**Naval Research Laboratory**
Stennis Space Center, MS 39529-5004

NRL/FR/7441--95-9633

# Compressed Aeronautical Chart Processing Operator's Manual

STEPHANIE A. MYRICK
MAURA C. LOHRENZ
J. MICHELLE MEHAFFEY
PERRY B. WISCHOW
MARLIN L. GENDRON

*Mapping, Charting, and Geodesy Branch*
*Marine Geosciences Division*

LANCE M. RIEDLINGER

*Planning Systems Incorporated*
*Slidell, LA*

May 6, 1998

Approved for public release; distribution unlimited.

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OBM No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | May 6, 1998 | Final |

**4. TITLE AND SUBTITLE**

Compressed Aeronautical Chart Processing Operator's Manual

**5. FUNDING NUMBERS**

*Job Order No.* 574-M09-608
*Program Element No.*
*Project No.*
*Task No.*
*Accession No.*

**6. AUTHOR(S)**

Stephanie A. Myrick, Maura C. Lohrenz, J. Michelle Mehaffey,
Perry B. Wischow, Marlin L. Gendron, and Lance Riedlinger*

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Naval Research Laboratory
Marine Geosciences Division
Stennis Space Center, MS 39529-5004

**8. PERFORMING ORGANIZATION REPORT NUMBER**

NRL/FR/7441--95-9633

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

National Imagery and Mapping Agency
12100 Sunset Hills Road
Reston, VA 22090-3221

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**

*Planning Systems Incorporated, 115 Christian Lane, Slidell, LA

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release; distribution unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** *(Maximum 200 words)*

The Naval Research Laboratory (NRL) Map Data Formatting Facility (MDFF) has developed an efficient data processing system that compresses National Imagery and Mapping Agency (NIMA) Equal Arc-Second Digitized Raster Graphics (ADRG) data into the Navy-specified Compressed Aeronautical Chart (CAC) database. In September 1995, the NRL MDFF team transitioned the CAC processing system (CPS) to NIMA. Documentation that was distributed with the system included *MDFF HELP* (both on-line and hard-copy versions), the *MDFF HELP Library Design Document*, and the *CAC Processing Operator's Manual* (this document).

This manual is intended for users of the CPS. It contains all the information required to create a CAC Optical Disk Image, including descriptions of all CAC processing phases, intermediate processing steps (within these phases), and operations and maintenance functions (e.g., utility software, significant directories, and files).

**14. SUBJECT TERMS**

ADRG, CAC, CDROM, CPS, Chart_Control, CHART_STATUS.DAT, Core, NIMA, INITIALIZATION, ISO-BUILD, PASS1, PASS2, PASS3, Process_Chart, Processing Thread, Segment, TS, WRAPUP

**15. NUMBER OF PAGES**

137

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | Same as report |

## CONTENTS


1.0 INTRODUCTION ..... 1

1.1 Background ..... 1
1.2 Overview of this Report ..... 1
1.3 Conventions Used in this Report ..... 2
1.4 Chart_Control Menu Conventions ..... 2

2.0 PROCESSING OVERVIEW ..... 3

2.1 Background ..... 3
2.2 CAC Driver Programs: Chart_Control and Process_Chart ..... 4
2.3 Processing Threads ..... 4
2.4 CPS Processing Phases ..... 4
2.5 System Description and Control ..... 6

3.0 OPERATION AND MAINTENANCE ..... 9

3.1 Processing Thread Operations ..... 9
3.2 Symbol Definitions ..... 11
3.3 Logical Name Definitions ..... 11
3.4 Utility Software ..... 11
3.5 Directories and Files ..... 14
3.6 Queue Management ..... 18
3.7 Disk Storage ..... 21
3.8 File Backup ..... 21

4.0 CAC PROCESSING INITIALIZATION ..... 23

4.1 Introduction ..... 23
4.2 Common Beginning Protocol ..... 24
4.3 Initialization for a New CAC ..... 25
4.4 Initialization for a CAC Update ..... 27
4.5 Error Recovery ..... 29

5.0 PASS1 PROCESSING ..... 30

5.1 Introduction ..... 30
5.2 Copying ADRG Data from CDROM ..... 31
5.3 Downsampling ..... 35
5.4 Color and Spatial Compressions ..... 37

5.5 Legend Processing ..... 39
5.6 Review Forms ..... 40
5.7 Other Chart_Control Options ..... 40

6.0 PASS2 PROCESSING ..... 46

6.1 Introduction ..... 46
6.2 Segment Classification: Core and Edge Segments ..... 47
6.3 PASS2_BUILD_SEGMENTS_FILE ..... 48
6.4 PASS2_BUILD_ODI_BOUNDS_FILE ..... 49
6.5 PASS2_DEFINE_TRIM_BOUNDS ..... 50
6.6 PASS2_TRIM_SEGMENTS ..... 51
6.7 PASS2_FIND_EDGEFILL_CDROMS ..... 51
6.8 PASS2_PROCESSING ..... 51
6.9 PASS2_FAUX_DOWNSAMPLE ..... 53

7.0 PASS3 PROCESSING ..... 53

7.1 Introduction ..... 53
7.2 PASS3_VERIFY_DS_SEGMENTS ..... 54
7.3 PASS3_BUILD_FILLED_PROCS ..... 56
7.4 PASS3_SUBMIT_FILLED_PROCS ..... 56
7.5 PASS3_COMPRESS_FILLED ..... 56
7.6 PASS3_BUILD_UNFILLED_PROCS ..... 56
7.7 PASS3_COMPRESS_UNFILLED ..... 56
7.8 PASS3_PROCESSING ..... 57
7.9 PASS3 Error Recovery ..... 57

8.0 WRAPUP PROCESSING ..... 60

8.1 Preliminary Considerations ..... 61
8.2 WRAPUP Steps and Menu ..... 62

9.0 ISO IMAGE PROCESSING ..... 63

9.1 Preliminary Considerations ..... 63
9.2 ISO Image Build and Copy ..... 64

10.0 REFERENCES ..... 66

APPENDIX A — Glossary of Terms ..... 67

APPENDIX B — Utility Software ..... 71

APPENDIX C — CAC Processing Check List ..... 127

APPENDIX D — CAC Processing System Startup ..... 129

APPENDIX E — CAC Processing System (CPS) Acceptance Test Procedures ..... 131

# COMPRESSED AERONAUTICAL CHART PROCESSING OPERATOR'S MANUAL

## 1.0 INTRODUCTION

### 1.1 Background

The Naval Research Laboratory (NRL) Map Data Formatting Facility (MDFF) has developed an efficient data processing system that compresses National Imagery and Mapping Agency (NIMA) Equal Arc-Second Digitized Raster Graphics (ADRG) data into the Navy-specified Compressed Aeronautical Chart (CAC) database. Much of the software in this system was originally developed by Honeywell, Inc., under contract to McDonnell Douglas Aircraft Co., as part of an airtask with the AV-8B program office at the Naval Air Systems Command (NAVAIR). From 1995, programmers at the NRL MDFF made significant modifications and enhancements to the original system, which considerably accelerated the production of the CAC database.

In September 1995, the NRL MDFF team transitioned the CAC processing system (CPS) to NIMA. Documentation that was distributed with the system included *MDFF HELP* (both on-line and hard-copy [1] versions), the *MDFF HELP Library Design Document* [2], and the *CAC Processing Operator's Manual* (this report).

### 1.2 Overview of this Report

This report is intended for users of the CAC CPS. It contains all the information required to create a CAC optical disk image, including descriptions of all CAC processing phases, intermediate processing steps (within these phases), and operations and maintenance functions (e.g., utility software, significant directories, and files).

This report describes the CPS from the perspective of an operator who processes source ADRG data into CAC data installments. It begins with an overview of CAC processing, including common acronyms and terminology, and a brief description of the CAC processing phases. Next, a section on Operations and Maintenance describes the basic computer system operations with which a CAC processing operator must be familiar. Included in this section are symbol and logical name definitions, directory and file structures, queue management, and related topics. The next five sections detail each phase of the CPS: INITIALIZATION, PASS1, PASS2, PASS3, WRAPUP, and ISO-BUILD. Appendices provide supplemental information: App. A contains a glossary of terms used within this report, and App. B contains a listing and description of utility software. App. C contains a CAC processing checklist, App. D contains CAC processing system startup procedures, and App. E contains CAC processing system Acceptance Test Procedures.

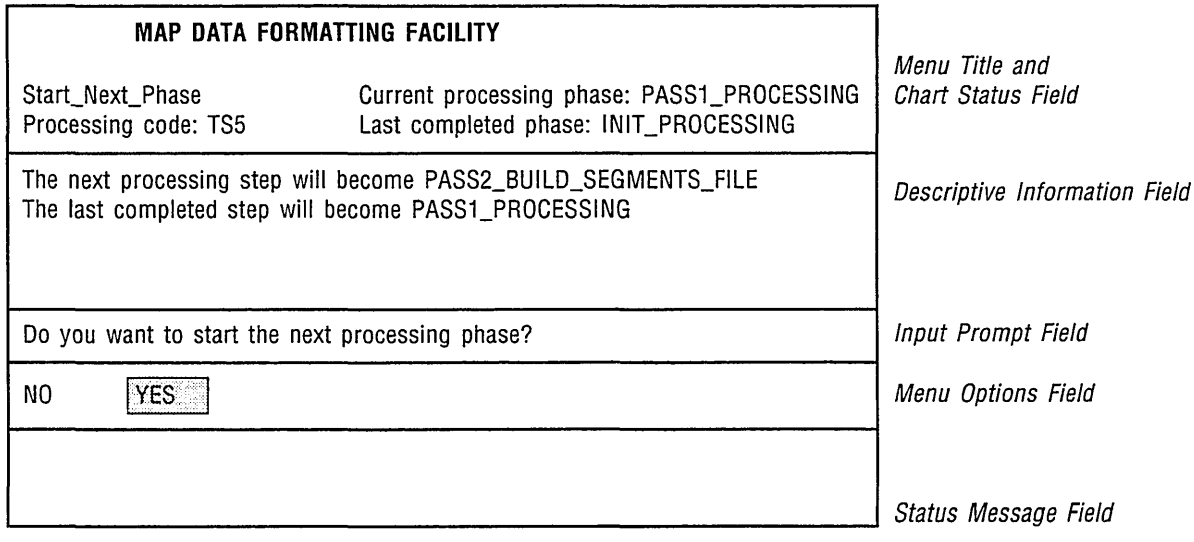
The reader is assumed to be familiar with Virtual Memory System (VMS) operations, including directory structure and hierarchy, file naming conventions, and the use of logical names and symbols. For more information about VMS, please refer to the VMS users manuals distributed with the CPS.

### 1.3 Conventions Used in this Report

| | |
|---|---|
| <key> | A specific key on the keyboard, such as <Return>, <Shift>, <Ctrl>, etc. |
| ^Z | A control character sequence: ^Z equates to holding down <Ctrl> while simultaneously pressing <Z>. |
| *italics* | Introduces a major term. Subsequent usage is non-italicized. Major terms are included in the Glossary (App. A). Also used for commands where the software name is italicized and in lower case. |
| <Return> | A carriage return terminates all commands except <Ctrl> sequences (e.g., ^Z). |
| *large italics* | Commands are depicted with a larger point italic font. For emphasis and clarity commands appear on one line of text and the VAX DCL prompt "$" is not shown. |
| **boldface** | User input (case insensitive; see following description). |
| UPPER/lower case | User input is not case sensitive, i.e., any user responses may be entered in upper and/or lower case. However, this report adheres to the following case conventions: |
| UPPERCASE | Acronyms, file names, directory names, logical names, symbol names, queue names, and processing phases and steps. |
| Title_Case | CAC production software names (e.g., Chart_Control, Process_Chart) and menu options (e.g., Validate_Readers) |

### 1.4 Chart_Control Menu Conventions

Chart_Control provides processing phase-specific menus. Each menu displays five information fields, as depicted and described below.



- *Menu Title and Chart Status Field:* displays menu title, processing thread name, and current processing status values.
- *Descriptive Information Field:* displays information specific to the current processing step. For example, in Fig. 1 (Start_Next_Phase Menu), information is provided about the next step.
- *Input Prompt Field:* displays input prompts.
- *Menu Options Field:* displays the options available for the current menu. Selected options are depicted with a gray-shaded frame.
- *Status Message Field:* displays various status messages.

## 2.0 PROCESSING OVERVIEW

### 2.1 Background

NIMA digitizes paper aeronautical charts into the ADRG database and distributes that data on CDROM. The CPS inputs ADRG CDROMs and compresses that data into the CAC format. One CAC CDROM (hereafter referred to as a "CAC") contains between 60–300 individual charts (depending on chart scale), or up to 65 compressed ADRG CDROMs. Each CAC contains data from a single chart series and scale.

ADRG data are stored in the Equal Arc-Second Raster Chart (ARC) system and CAC in the Tessellated Spheroid (TS) Map Projection System. ARC and TS are similar methods of segmenting chart data into manageable pieces for efficient storage, manipulation, and display. During the conversion to CAC, data are transformed from the ARC system to TS.

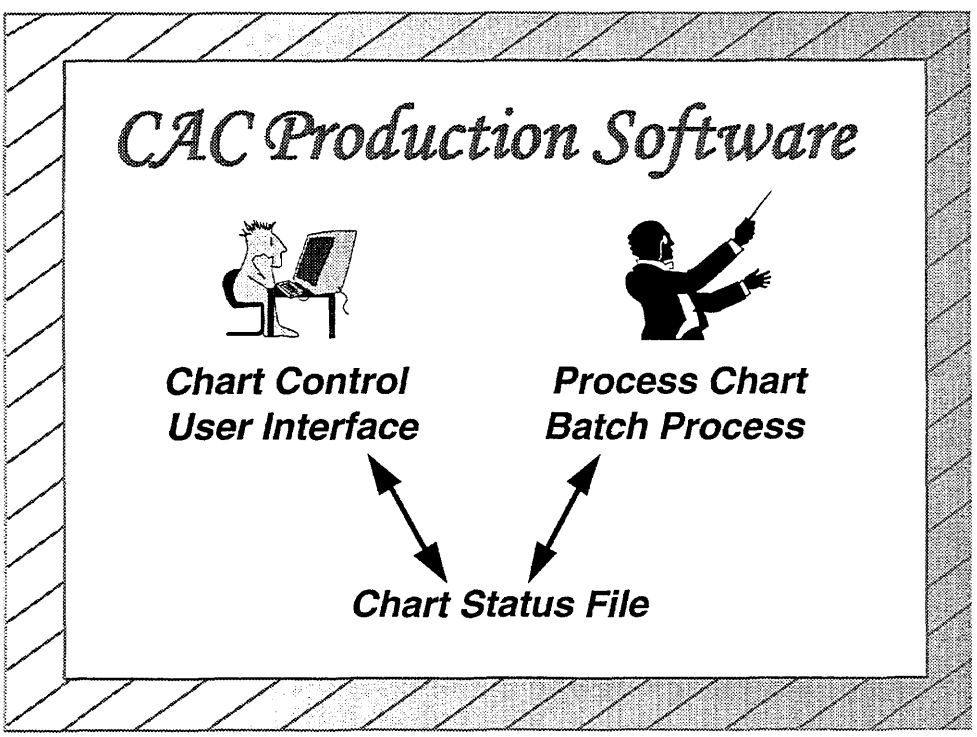



Fig. 1 — Chart_Control and Process_Chart functionality

ARC and TS both provide a rectangular coordinate and projection system at any scale for the entire Earth ellipsoid. Based on this convention, a pair of *row* and *column* values correspond to a rectangular area of coverage. In the TS system, each rectangular area of coverage is known as a *segment.*

The TS system divides the Earth surface into five latitudinal *zones*: two polar zones (North Polar and South Polar) and three nonpolar zones (North Temperate, Equatorial, and South Temperate).

The reader is referred to the CAC product Specification [3] and the TS Specification [4] for more information about the CAC data format and TS, and NIMA MIL-A-89007 (the ADRG Military Specification [5]) for more information about ADRG and ARC.

### 2.2 CAC Driver Programs: Chart_Control and Process_Chart

Two driver programs coordinate the entire CPS: Chart_Control and Process_Chart (Fig. 1). These drivers invoke other subroutines and programs to perform various processing tasks. Both use the CHART_STATUS.DAT file for retrieving and maintaining current processing information.

Chart_Control is a user-interface program that presents menus specific to each processing phase. Chart_Control also provides the mechanism to control processing; it is used to advance control to the next processing step, or to restart processing from a previous step.

Chart_Control must be executed from a predefined CAC user account to have access to the necessary privileges and logical name definitions.

Process_Chart coordinates all the processing tasks and systematically calls appropriate software routines to perform those tasks. For example, in the *PASS1* phase, Process_Chart invokes the appropriate subroutines for PASS1 processing (e.g., copying ADRG data from CDROM and performing initial data compression tasks).

Chart_Control and Process_Chart share information through the CHART_STATUS.DAT file. The CHART_STATUS.DAT file is the repository for all information relevant to a processing thread.

### 2.3 Processing Threads

The CPS can complete several CACs (of the same or different scales) simultaneously through the use of "*processing threads.*" One processing thread is established for each new CAC to be processed. Establishing a processing thread involves allocating resources and establishing a unique file name for a given CAC. The status of any CAC-in-process can be precisely determined by examining the state of its processing thread. A processing thread is created via Chart_Control, as described in the INITIALIZATION section of this report.

### 2.4 CPS Processing Phases

CAC processing is comprised of five phases: INITIALIZATION, PASS1, PASS2, PASS3, WRAPUP, and ISO_BUILD.

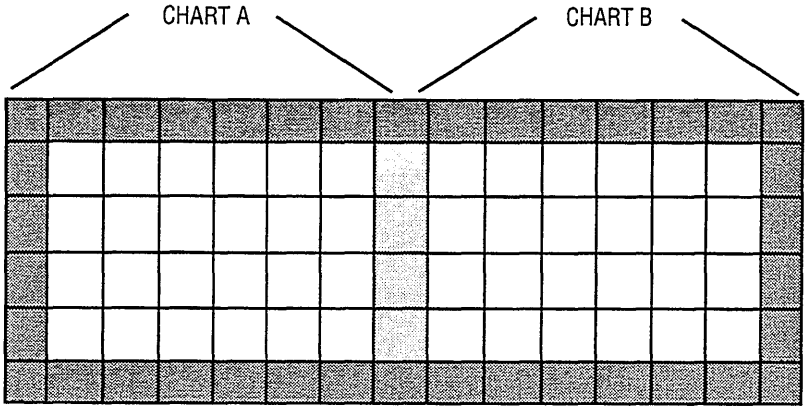



Fig. 2 — Core and edge segments of two bounding charts. Core segments (depicted in white) cover the innermost portion of a chart. Edge segments (depicted in dark gray) contain portions of a chart's edges. Edge segments from bounding charts may be combined to form a full segment (depicted in light gray).

INITIALIZATION defines a new processing thread and allocates system resources. During this phase, Chart_Control prompts the operator for the processing thread name, the chart scale of the data to be processed, and other relevant information.

PASS1 begins the actual data processing by copying ADRG data from CDROM onto a local disk and compressing *core segments.* As shown in Fig. 2, core segments cover the innermost portions of a chart and *edge segments* contain portions of a chart's edges.

Edge segments usually contain data from more than one chart (and, therefore, more than one ADRG CDROM), so edge segments are not processed until all available ADRG data (for the current processing thread) have been copied to disk and are ready to be compressed. For this reason, edge segments are handled in the next phase (PASS2) after all ADRG data have been copied and all core segments have been processed. Further details about processing (i.e., downsampling and compression) are explained in Sec. 5.3.

PASS2 fills each edge segment with all available ADRG data for that segment. Edge segments that are completely filled with data (usually from more that one chart) are referred to as *filled* edge segments; all other edge segments in the thread are referred to as *unfilled* edge segments. Unfilled edge segments are partially filled with data (they are not empty). PASS2 also defines the exact area of coverage for the CAC Optical Disk Image (ODI). All segments that lie outside the defined area of coverage are *trimmed* before the ODI is built. For example, Fig. 3 shows the CAC ODI bounds to include those segments that are defined within the rectangle (Australia and Tasmania). All other segments (New Zealand) would not be included as part of this CAC ODI and, as a result, are trimmed.

PASS3 compresses the filled and unfilled edge segments. Upon completion of PASS3, all the data to be included in this current CAC have been compressed.

WRAPUP creates all necessary CAC header files to be included in the ODI, including files that store overall CAC CDROM coverage, color palette coverages, area source, and area Distribution

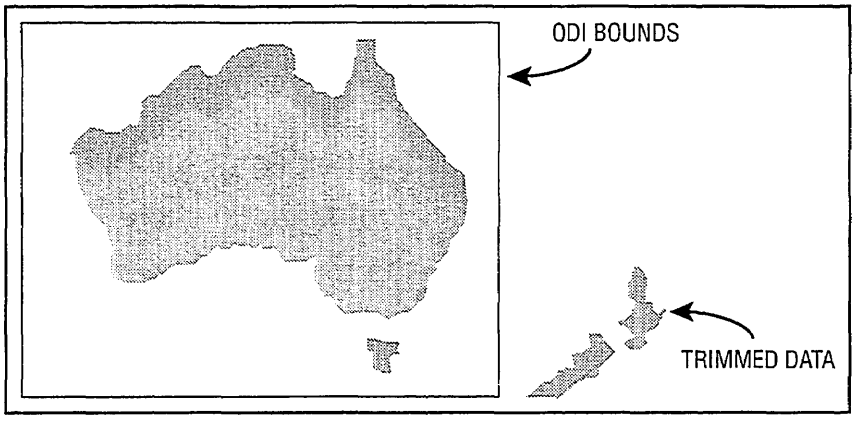



Fig. 3 — Example of trimming during PASS2 CAC area of coverage definition. All data outside the ODI bounds are trimmed prior to building the ODI.

Rectangle (DR) information. WRAPUP also deletes any temporary "scratch" files and other extraneous files.

ISO IMAGE BUILD processing is the final CPS phase. The ODI is a "snapshot" of the entire file system (including directories and data files) to be mastered on CDROM. The end-user systems require the ODI to be written in the ISO 9660 standard format. Once the ODI has been built in the ISO format, it is copied to tape and sent to the CDROM mastering facility. An overview of CPS processing phase functionality is shown in Fig. 4. A checklist is provided in App. C to ensure that processing progresses normally.

## 2.5 System Description and Control

This section describes the CPS hardware configuration and addresses issues relevant to management of computer system resources. The CPS uses Digital Equipment Corporation's (DEC) highly reliable and efficient AXP computer architecture. The CPS runs under the VMS operating system to provide a user-friendly and robust environment for data processing.

Instructions for powering up and starting the system (hardware and software components) are provided in App. D. The Acceptance and Test Procedures (provided as part of this system deliverable) augment this information and are provided in App. E.

### *2.5.1 Hardware Configuration*

The CPS consists of a DEC 4720 AXP computer with two 190 Mhz CPUs, each of which has 512 Mb of memory. Additional storage and peripherals include one 2-Gb system disk, eight 2-Gb processing disks, one system console; six CDROM readers; one 8-mm tape unit; one Digital Linear Tape unit (DLT); one Tektronix color printer; one laser printer; and three VXT2000+ X-Window terminals (Fig. 5). Two of the CDROM readers are in the computer cabinet and the other four are in an external pedestal cabinet. The 8-mm tape drive is used to write the final International Standards Organization (ISO) image for transport to the mastering facility and to backup processing threads. The DLT drive is used to install software updates and to create system and CPS backups.

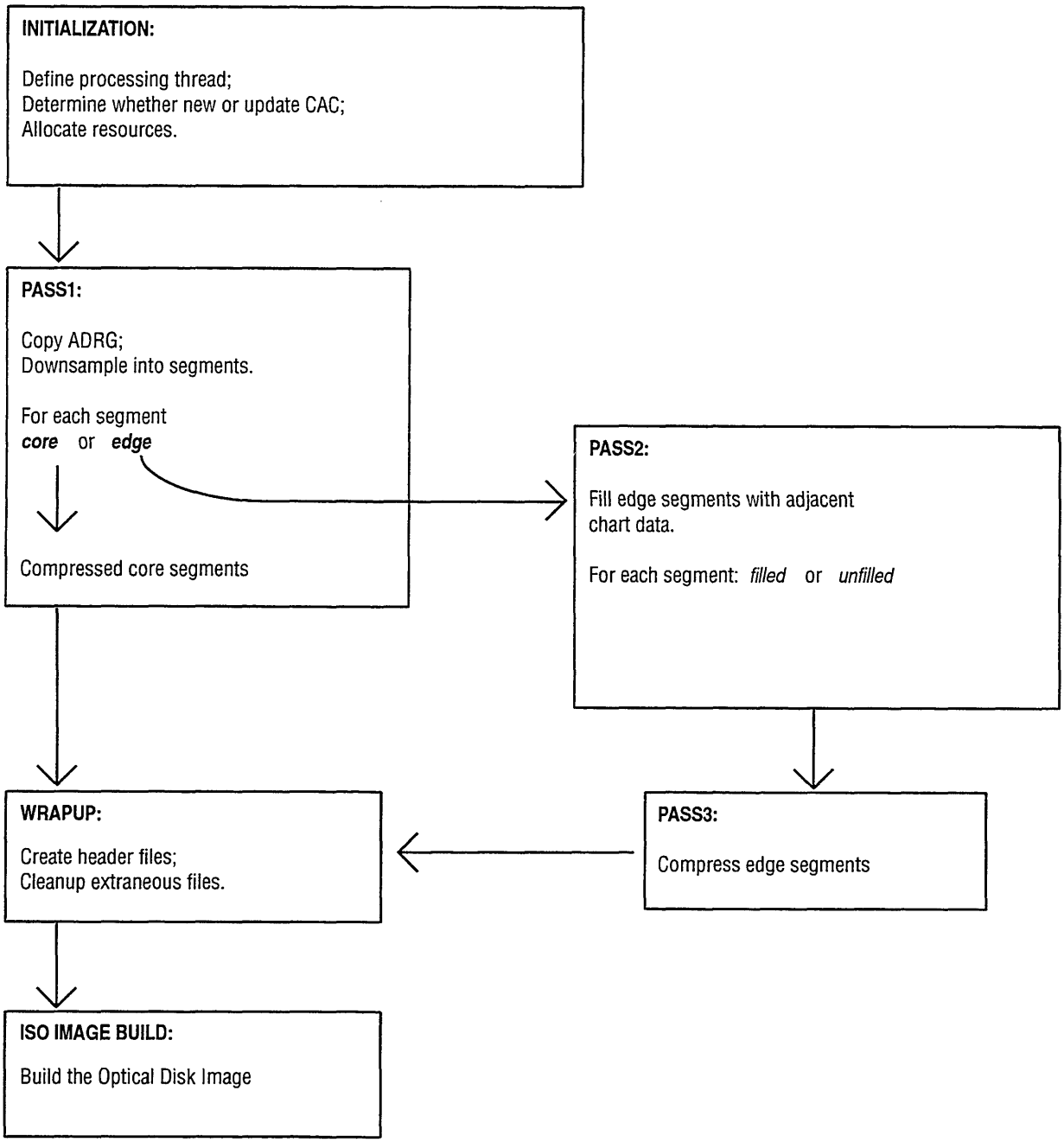



Fig. 4 — CAC Processing Overview and Functionality provided through Chart_Control and Process_Chart

### *2.5.2 Software Configuration*

The CPS runs with the DEC VMS operating system. Since VMS uses a high-level control language, it provides an easy-to-learn and user-friendly operating environment. VMS affords the creation and use of logical names and symbols that have been constructed to meet CPS requirements. The on-line *help* facility provides a thorough description of the command language and VMS. An additional on-line help facility, MDFF HELP, provides information specifically covering CAC production, the CPS, and other relevant information.

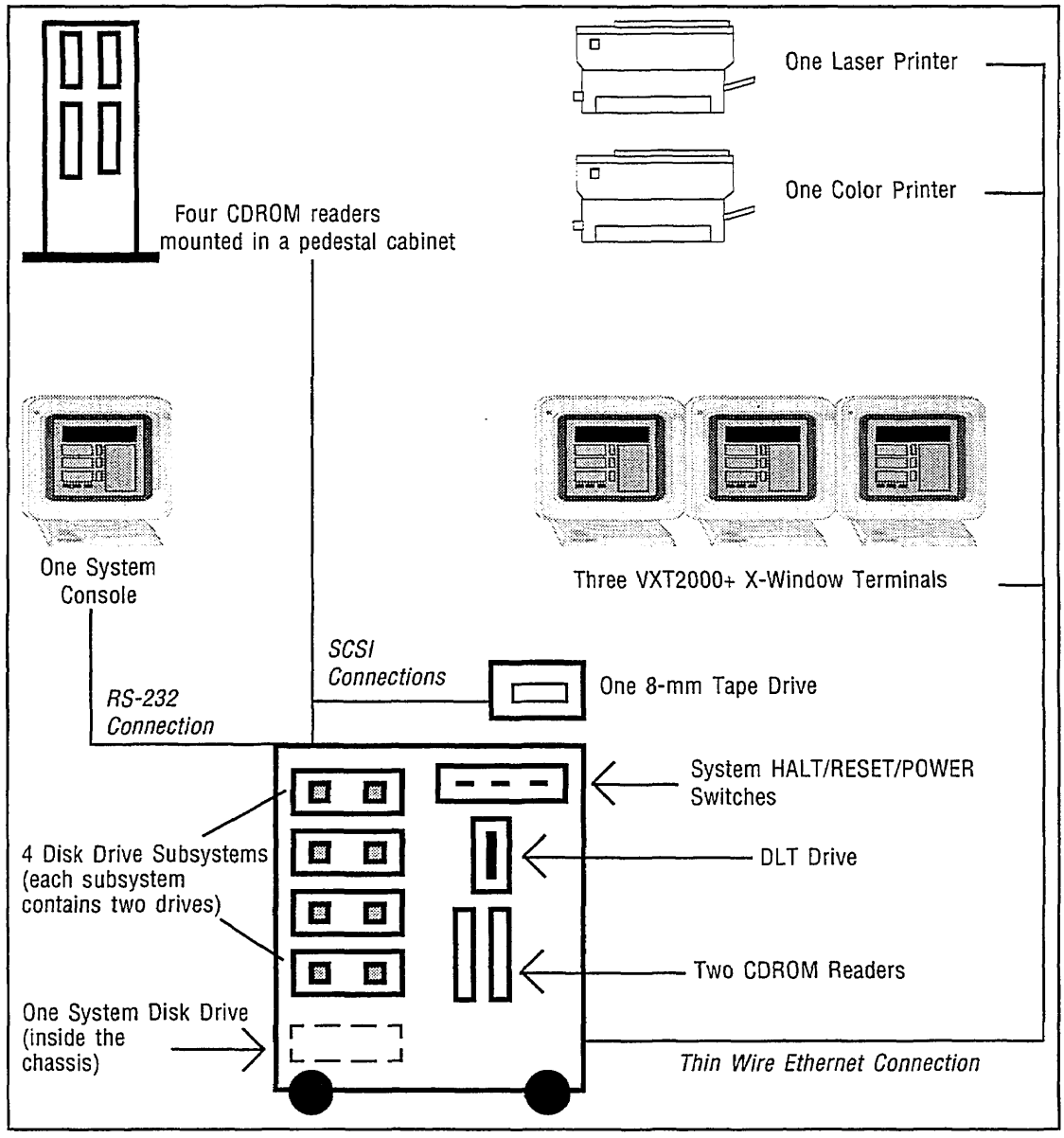



Fig. 5 — CPS hardware configuration

### 2.5.3 *Operator Console Messages*

Operator Console Messages (OPCOM) are used to monitor the CPS and relay all error and status messages (for additional information, type *help reply/enable*). The CPS sends notices to terminals that are enabled to receive OPER5 OPCOM notices and the system console receives all OPCOM notices by default. Figure 6 shows common commands for managing OPCOM information.

Much of the OPCOM text is written to the SYS$MANAGER:OPERATOR.LOG file. Since this file is often modified, many versions of this file are created. This file needs to be periodically purged to avoid using excessive disk space.

| COMMAND | ACTION |
|---|---|
| *reply/enable* | Receive all OPCOM messages at terminals other than the default OPCOM terminal |
| *reply/enable=OPER5* | Receive only CPS notices |
| *reply/disable* | Turn off all OPCOM notices |
| *reply/disable=OPER5* | Turn off only CPS notices |
| *reply/enable*, followed by *reply/status* | Clear for any outstanding notices |
| *reply/to*=notice_# required_response | Respond to an outstanding notice |
| *reply/abort*=notice_#_to_cancel | Delete an outstanding notice |

Fig. 6 — Common commands for managing OPCOM information

### *2.5.4 DTR Database Compaction*

The CPS uses the DEC DATATRIEVE program to manage databases of ADRG CDROM and CPS audit trail information. New records are added to the DATATRIEVE databases continuously, so files can become badly fragmented, which can significantly degrade database access time. Use the following command to periodically defragment and create new database files (with the same names):

*$set def MDFF_SYSTEM:[DATABASES]*
*$@DEFRAGMENT_DAT_FDL.COM.*

Processing should be HALTED and the DTRLIB files *should not be accessed* when the DATATRIEVE database files are being rebuilt.

## 3.0 OPERATION AND MAINTENANCE

### 3.1 Processing Thread Operations

*Processing threads* are used to differentiate between concurrent CAC ODI builds, as mentioned in Sec. 2.3. A processing thread is a logical name that defines all the resources associated with a particular CAC ODI build, including directory names, device names, and queue names. This section describes common processing thread operations including definition, restoration, and deletion.

### *3.1.1 Processing Thread Definition*

Processing threads are created during the INITIALIZATION (Sec. 4.0) and use a standard three-character code:

- The first character distinguishes the data set type; for CAC, an "A" *must be used* (for aeronautical chart).
- The second character identifies the scale of the data set [(0)-1:50K, (1)-1:100K, (2)-1:250K, (3)-1:500K, (4)-1:1M, (5)-1:2M), and (6)-1:4M]. *The only valid* values to use are between 0 and 6.
- The third character is a sequence identifier [(A)-first, (B)-second, etc.]. This is used to delineate simultaneous CAC ODI builds when both charts are at the same scale.

For example, the processing thread *A4A* denotes a CAC ODI build for an (*A*) aeronautical chart at the (*4*) operational navigation chart (ONC) (1:1M) scale with sequence identifier (*A*).

Since multiple CACs can be processed concurrently, a processing thread must be defined to access chart data that is associated with a particular CAC ODI build. The *current processing thread* is the thread that was *most recently* defined. Use the following utility to define an existing processing thread:

*thread* xxx (xxx = processing thread name).

### *3.1.2 Processing Thread Restoration*

Processing thread restoration is required *only* in the event of a catastrophic system failure or if the processing thread becomes so corrupted that the user cannot trust the validity of the processed data.

The following procedure is used to restore a processing thread:

- Delete the existing thread (Sec. 3.1.3).
- Re-initialize the thread without the update option (even if the thread being restored is an update).
- Execute *@MDFFEXE:SUBMIT_RESTORE_THREAD.COM*. The thread name being restored and the tape drive containing the backup (8 mm) will be prompted. This program will submit a thread-specific restore procedure to the MDFF_BATCH queue.
- Run *sqmdff* to display the queue status. A job name of RESTORE_xxx should be displayed (where xxx is the processing thread name).

When the restore operation completes, Chart_Control notifies the user via a mail message. Processing can continue from the phase/step displayed with LIST_CHART_STATUS or control may be returned to a previous phase/step using the RESTART_PREVIOUS_PHASE menu option in Chart_Control (Sec. 5.7.3.3).

### *3.1.3 Processing Thread Deletion*

A processing thread should only be deleted if one of the following conditions is met:

- The thread is completed, the ISO image has been verified and the CAC CDROM has been pressed and verified.
- There is a change in requirements such that the data is no longer needed. Note that if a backup is done before the deletion, then the thread may be restored at a future date with no loss of data or processing time.
- There is insufficient disk space for the current priority.

To delete a processing thread execute the following command file:

*@MDFFEXE:SUBMIT_DELETE_THREAD.COM.*

The thread name being deleted will be prompted. This will submit a thread-specific delete procedure to the MDFF_BATCH queue. Type sqmdff to display the queue status. A job name of

DELETE_xxx should be displayed (where xxx is the processing thread name). A mail message is sent when the delete is completed.

An alternative to deleting a processing thread is to render the thread temporarily inactive (i.e., it will be undefined after a system reboot). To do this, rename the file MDFF_SYSTEM:[COMMON] DEFINE_xxx_LOGICALS.COM to MDFF_SYSTEM:[COMMON]DEFINE_xxx_LOGICALS.COM_ OLD (where xxx is the processing thread name).

## 3.2 Symbol Definitions

Symbols are used to replace lengthy commands for clarity and brevity. For example, the symbol "thread" is used instead of the command "@MDFFEXE:USE_ADRG_LOGICALS.COM." All CPS symbols are *automatically* defined by the command file MDFF_SYMBOLS.COM when the user logs onto the system.

## 3.3 Logical Name Definitions

There are two sets of CPS logical name tables: a *general table* to control general functions (e.g., print queues and interprocess communications) and a *thread-specific table* to control the characteristics and functions of a processing thread (e.g., compression queue assignments and concurrent ADRG CDROM processing constraints).

General logical names are defined in the MDFF logical name table, which can be listed using the following command:

*show logical/table* = **MDFF**.

Thread-specific logical names are listed using the following command:

*show logical/table* = **xxx** (**xxx** is the processing thread name).

Most logical names are not modifiable. However, some may be modified (Fig. 7) to accommodate changing processing requirements.

## 3.4 Utility Software

Utilities that monitor and maintain a processing thread are identified in this section (on-line information is also available in MDFF HELP). More detailed information about most of these utilities is provided in App. B.

### *3.4.1 CAC_QC*

This program provides quality control functions, including review of data being processed in a processing thread, review of data from CAC CDROM, and review of CAC color palettes. A full program description is provided in App. B.

### *3.4.2 Display_ADRG_X*

This program is used to view ADRG data from CDROM, to select ADRG data for CAC color palette creation, and to check for source data flaws during CAC processing (e.g., unreadable tiles, missing data, and corrupted data). A full program description is provided in App. B.

| | |
|---|---|
| ***Modifiable General MDFF Logical Names*** | |
| MASTERING FACILITY | An identifier for the facility used to master the CAC CDROM where 0 = Discovery Systems, 1 = 3M, 2 = NIMA, St. Louis, 3 = Digital Audio Disk Corporation |
| MDFF_PRINT_QUEUE | The queue that prints text output |
| OPEN_SEGMENT_INTERVAL | Time in seconds between attempts to open a downsampled segment file that has been locked by another process |
| OPEN_SEGMENT_TRIES | Number of attempts to open a downsampled segment file that has been locked by another process, before aborting |
| PROCESS_CHART_INTERVAL | Number of seconds Process_Chart hibernates |
| PROCESS_PALETTE_INTERVAL | Number of seconds Process_Palette hibernates |
| TIMETAG_INTERVAL | Time (hh:mm) format between Process_Chart/Process_Palette time tag OPCOM messages |
| ***Modifiable Processing Thread-Specific Logical Names*** | |
| CODEBOOK_QUEUES | Execution queues that compression procedures are submitted to |
| CONCURRENT_CDS | Number of ADRG CDROMs that can be in the process of copying or downsampling at one time |
| MDFF_READERS | CDROM reader names that are presented in the INITIALIZATION Reload menu, the PASS1 Validate menu, and the PASS2 Validate/Select menu |

Fig. 7 — Modifiable logical names

### *3.4.3 Display_Segments_X*

This program is used to monitor the progress of CAC processing and to facilitate processing recovery after system failure. Capabilities include display of ADRG CDROMs that have been selected for processing and a wire-frame diagram showing downsampled and compressed segments. A full program description is provided in App. B.

### *3.4.4 Dump_CAC_Files*

This program is used to list the contents of files with the extension. DAT that were built during the WRAPUP phase of processing. A full program description is provided in App. B.

### *3.4.5 List_Chart_Status*

This utility lists values within the contents of the CHART_STATUS.DAT file for the currently defined processing thread and it is invoked using the following command:

*list_chart_status.*

A hardcopy of CHART_STATUS.DAT is obtained by using the print qualifier:

*list_chart_status/print* = **printer queue name.**

If a print queue name is omitted, the queue defined as MDFF_PRINT_QUEUE is used by default. Additional information is provided in App. B.

### *3.4.6 List_Codebook_Status*

This utility lists the contents of a CODEBOOK_STATUS.DAT file. The only required input to this utility is an *MDFF sequence number.* Sequence numbers are assigned for every ADRG CDROM that has been selected for processing, and they are listed (with their associated ADRG CDROM) by the List_Chart_Status utility.

The following command invokes the List_Codebook_Status utility for the ADRG CDROM with sequence number 5001:

*list_codebook_status*/cd = **5001.**

This utility can also be used for color palette codebooks. The following command invokes the List_Codebook_Status utility for CAC color palette number 130 and ADRG CDROM with sequence number 5445:

*list_codebook_status /pa* = **130** */cd* = **5445.**

Additional qualifiers and other information are provided in App. B.

### *3.4.7 Plot_ODI*

This program is used to determine which areas of coverage have already been processed into CAC. PLOT_ODI is a planning tool for selecting ADRG CDROMs for CAC processing. Once a chart scale has been selected, all the ADRG CDROMs that are registered in the MDFF DATATRIEVE library (for the designated scale) are identified and their areas of coverage displayed. A World Vector Shoreline overlay is optionally available. A full program description is provided in App. B.

### *3.4.8 Sqcode*

This utility is used to list current generic batch and codebook compression queue assignments. A full utility description is provided in Sec. 3.6.2 and also in App. B.

### *3.4.9 Sqmdff*

MDFF_BATCH queue activity is monitored by the Sqmdff utility:

*Sqmdff* (no parameters).

An example of Sqmdff output is provided below:

Batch queue MDFF_BATCH, available, on ZOMBIE::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 897 | MDFF_CONTROL | SYSTEM | Executing |
| 351 | A2A_PROCESSCHRT | CAC | Executing |
| 534 | A2C_PROCESSCHRT | CAC | Executing |

### *3.4.10 Thread*

Since multiple CACs can be processed concurrently, a processing thread must be defined to access chart data associated with a particular CAC ODI build. The *current processing thread* is

the thread that was *most recently* defined. Use the Thread utility to define an existing processing thread:

*thread* xxx (xxx = processing thread name).

### 3.5 Directories and Files

The following sections describe significant directories and files used during CAC processing. CPS directories are either thread-specific (i.e., they only contain files associated with a particular processing thread), or general (i.e., their contents are not confined to any particular processing thread).

#### *3.5.1 Thread-Specific Directories*

Thread-specific directories include: MDFF_SCRATCH, CHART_SEGMENTS, CHART_ODI_DISK, TRIMMED_DS, and TRIMMED_CS.

The MDFF_SCRATCH directory is a repository for data files, command procedures, log files, and ADRG CDROM subdirectories used in CAC processing. Figure 8 provides a sample of the contents of this directory.

The CHART_SEGMENTS directory stores all downsampled segments. The CHART_ODI_DISK directory is the root CAC directory for a given CAC-in-process: within this directory are subdirectories that store all the compressed segments for the thread including "MAPx" (where x indicates the map scale) that contains the actual compressed segment data, and "ID," which contains associated header files. For example, CHART_ODI_DISK:[MAP3] is used for Tactical Pilotage Chart (TPC) data (the 3 indicates TPC scale).

Two directories store data that are trimmed from the CAC ODI during PASS2: TRIMMED_DS, which contains the trimmed downsampled files, and TRIMMED_CS, which contains the trimmed compressed files.

#### *3.5.2 General Directories*

General directories are not limited to any particular processing thread. Hence, they contain CPS software, copied ADRG data files, and ISO image files.

| FILE NAME | DESCRIPTION |
|---|---|
| A6A_BLDPASS1_02.COM | Builds a PASS1 codebook compression procedure |
| A6A_BLDPASS1_02.LOG | PASS1 codebook compression procedure log file |
| A6A_PROCESSCHRT.COM | Executes Process_Chart |
| A6A_PROCESSCHRT.LOG | Process_Chart execution log file |
| A6A_PROCLEGND01.COM | Processes legend data |
| A6A_PROCLEGND01.LOG | Log file for legend data processing |
| BACKUP_A6A_FILES.COM | Performs BACKUP operations |
| CD005001.DIR | ADRG CDROM subdirectory for sequence number 5001 |
| CHART_STATUS.DAT | Processing thread control file |

Fig. 8 — Contents of MDFF_SCRATCH directory

| DESCRIPTION | LOGICAL NAME | FULL PATHNAME |
|---|---|---|
| Command Procedures | MDFFCOM | MDFF_SYSTEM:[SOFTWARE.COM] |
| Executable Images | MDFFEXE | MDFF_SYSTEM:[SOFTWARE.EXE] |
| Object Code Files | MDFFOBJ | MDFF_SYSTEM:[SOFTWARE.OBJ] |
| Source Code Files | MDFFSRC | MDFF_SYSTEM:[SOFTWARE.SRC] |

Fig. 9 — MDFF_SYSTEM:[SOFTWARE] directories

The CPS is stored in subdirectories within the MDFF_SYSTEM:[SOFTWARE] directory. Logical names are always used to reference these subdirectories. For example, the logical name MDFFEXE references the MDFF_SYSTEM:[SOFTWARE.EXE] directory that contains all CPS executable images. Figure 9 lists these subdirectories with their logical names.

Data copied from ADRG CDROMs are stored in the following directories:

ADRGDIRxx:[FILES], where xx is a number 1 through 6 that corresponds to a CDROM reader.

ISO images are stored in the directory defined by the logical name ISO_IMAGES.

```
        Time of listing: 27-JAN-1995 10:45:53.42
              Processing thread: A6A

*** CHART_STATUS header ***

                ODI_TYPE:      CAC_ODI
     COMPRESSION METHOD:       NRL_COMP
               MAP_SCALE:      GNC (4M    )
       BUILD_START_TIME:       Wed Apr  6 11:38:24 1994
         BUILD_END_TIME:       Thu Jul  7 14:12:03 1994
      LAST_BACKUP_TIME:        Fri May 13 10:05:07 1994
        PROCESSING_STEP:       PASS1_PROCESSING
   LAST_COMPLETED_STEP:        INIT_PROCESSING
        REGRESSION_STEP:       NO_OP
     NUMBER_OF_ENTRIES:        9
     CDROMS_PROCESSED:         24
        CDS_IN_PROCESS:        0
          ODI_NUMBER:          000
          ODI_VERSION:         A
     MASTERING_FACILITY:       0
           SEGS_PER_CB:        1
       VERIFED_DS_SEGS:        FALSE
         RELOAD READER:
           NUM_PA_CBS:         0
  NUM_PA_CBS_COMPLETED:        0
         LIBRARY_UPDATE:       FALSE
PROCESS_ALL_LEGEND_DATA:       TRUE
         CLEAN_UP_FLAG:        FALSE
              ODI_SIZE:        0
```

Fig. 10 — Status_Header Listing (output of List_Chart_Status/ Header=Only utility)

### *3.5.3 CHART_STATUS.DAT File*

Each processing thread has a CHART_STATUS.DAT file associated with it. CHART_STATUS. DAT is the master control file since it contains all relevant information regarding processing. CHART_STATUS.DAT resides in the MDFF_SCRATCH:[000000] directory for the current processing thread. The file format is defined within the MDFFSRC:CHART_STATUS.INC file.

Information in the CHART_ STATUS.DAT file is organized into two parts: *header* information and ADRG CDROM information. Header information is stored within the *status_header* record and includes fields such as map scale, current processing step, ODI backup date, etc. Figure 10 provides a sample status_header listing that was printed using the List_Chart_Status utility with the/Header=Only qualifier.

Each ADRG CDROM that was selected for CAC processing is represented by a *status_record* in the

CHART_STATUS.DAT file. Each status_record includes the ADRG CDROM's NIMA stock number, MDFF sequence number, and current and last processing steps. Figure 11 illustrates abbreviated status_record entries that were printed using the List_Chart_Status utility.

The status_record and status_header together describe the current status of a CAC processing thread. A description of the status_header and status_record field mnemonics and values is provided in the file MDFFSRC:CHART_CONTROL_FLAGS.INC.

### *3.5.4 CODEBOOK_STATUS.DAT File*

The CODEBOOK_STATUS.DAT file logs information pertaining to the codebook and compression procedure build/compression steps for PASS1 and PASS3. Since there is one CODEBOOK_STATUS. DAT file for each ADRG CDROM that is processed during PASS1, every compact disk (CD) (i.e., CDROM) subdirectory (within MDFF_SCRATCH) will contain its own version of this file. The CD subdirectories also contain the command files that build the compression procedures. In PASS3, there is one codebook status file for each TS zone.

During data compression, the CODEBOOK_STATUS.DAT file is checked for successful and nonsuccessful codebook and compression procedure builds. Figure 12 provides a sample CODEBOOK_STATUS.DAT file listing that was produced using the List_Codebook_Status utility.

```
*** CHART_STATUS records ***
1 5001 ARC4 GNCXX01     NO_OP                 CD_COMPLETE
2 5002 ARC4 GNCXX02     BUILD_ADRG_CB_PROCS   PASS1_DOWNSAMPLE
3 5003 ARC4 GNCXX03     NO_OP                 CD_COMPLETE
6 5004 ARC4 GNCXX04     NO_OP                 CD_COMPLETE
7 5005 ARC4 GNCXX05     NO_OP                 CD_COMPLETE
8 5006 ARC4 GNCXX06     NO_OP                 CD_COMPLETE
9 5007 ARC4 GNCXX07     NO_OP                 CD_COMPLETE
```

Fig. 11 — Abbreviated status_record entries listing (output of list_chart_status utility)

```
_CD Number: 5001
*** List of CODEBOOK_STATUS file for CD#  5001,  28 entries ***

                Map  Pal          CB_To
CB#  Stat  Pass  Zone  Id    Use  MinRow  MinCol  MaxRow  MaxCol  Distort  Last_Seg
001  99    1     0     0000  0    00000   00000   00000   00000   338      0166410522
002  99    1     0     0000  0    00000   00000   00000   00000   353      0166572531
003  99    1     0     0000  0    00000   00000   00000   00000   292      0166734540
 .
 .
 .
025  99    1     0     0000  0    00000   00000   00000   00000   209      0169470692
026  99    1     0     0000  0    00000   00000   00000   00000   231      0169812711
027  99    1     0     0000  0    00000   00000   00000   00000   290      0169974720
028  0     0     0     0000  0    00000   00000   00000   00000   0        0000000000
```

Fig. 12 — Example of CODEBOOK_STATUS.DAT file

| STATUS | DESCRIPTION |
|---|---|
| 0 | No action taken |
| 1 | Procedure is finished, log file not yet read and logged |
| 99 | Successful completion |
| 77 | Compression failure |
| 88 | Failed Codebook build |

Fig. 13 — Codebook return values

The List_Codebook_Status utility also returns a status code for the ADRG CDROM. Figure 13 lists all the possible status codes and their descriptions. Normally, compression procedures with a status of 88 are automatically resubmitted by Process_Chart after all codebook entries have been flagged with either 99 or 88. However, abnormal events (e.g., a system crash) may prevent their resubmission. An ADRG CDROM currently flagged as having a process state "ADRG_CD_PROCS_SUMBITTED" in the CHART_STATUS.DAT file must have its command procedures submitted to codebook queues for normal processing to continue. Hence, it is *imperative* that the operator verify that these procedures have been submitted and are correctly listed in the queues. Use either the sqcode utility (Sec. 3.6.2) or the sqmdff command to show existing batch jobs associated with CAC processing.

Codebook entries with a status of 77 are not automatically resubmitted since they reflect errors that require operator assistance to be corrected. Once the error(s) have been corrected, these procedures can be completed with the Chart_Control CONTROL_OPTIONS menu RECOVER_CHART_BUILD option.

If no jobs are active, or if jobs are missing from the queue, then a RECOVER_CHART_BUILD operation is in order. This operation resubmits all incomplete jobs. Section 5.6.3.1 describes the actions required to perform a RECOVER_CHART_BUILD operation.

### *3.5.5 DS_SEGMENT.DAT and LAST_SEGMENT.DAT Files*

DS_SEGMENT.DAT files contain information about the individual segments that were downsampled from a given ADRG CDROM. This information includes the total number of downsampled, TS row number, TS column number, TS zone number, color palette number, pixel count (i.e., the number of pixels that were downsampled within the segment), compression codebook number, and flags pertaining to review, repair, *faux downsample* (Sec. 6.9), and segment-fill operations.

One DS_SEGMENT.DAT file exists for *every* ADRG CDROM that has been downsampled for CAC processing. These files are located in the corresponding MDFF_SCRATCH CDROM subdirectories. Hence, the DS_SEGMENTS.DAT file for ADRG CDROM number 5001 is located in the MDFF_SCRATCH subdirectory CD005001.DIR.

The List_DS_Segments utility lists the contents of a DS_SEGMENTS.DAT file. The only input required for this utility is the MDFF sequence number for the ADRG CDROM.

A LAST_DS_SEGMENT.DAT file contains the row, column, and zone numbers of the last segment downsampled. This file is located within each of the ADRG CDROM subdirectories of MDFF_SCRATCH. It is used to restart downsample procedures during the RECOVER_CHART_BUILD operation.

### *3.5.6 SEGMENTS.DAT File*

The SEGMENTS.DAT file lists information about *all the segments that comprise a CAC ODI* (whereas the DS_SEGMENTS.DAT file contains information about the segments from only one ADRG CDROM). This information includes the total number of segments that were processed and, for each segment, its row number, column number, zone number, palette number, pixel count (i.e., the number of pixels within the segment), compression codebook number, and status flags.

There is only one SEGMENTS.DAT file per CAC, which is stored in the MDFF_SCRATCH directory. Process_Chart creates this file at the completion of PASS1. Since it is first used in PASS2 for distinguishing segment types, SEGMENTS.DAT is fully described in the PASS2 Sec. 6.3.1 of this report. Once processing has finished and the CAC ODI is complete, SEGMENTS.DAT contains pertinent information about every segment that was processed during each phase.

## 3.6 Queue Management

### *3.6.1 MDFF_BATCH Queue*

The progression of CAC processing is monitored through batch queue activity, as well as status files and SEGMENTS.DAT files. MDFF_BATCH is a generic batch queue that handles all CAC processing activity except (codebook) compression procedures. MDFF_BATCH executes various command procedures to copy ADRG data from CDROM to disk, downsample the data into TS segments, and build other command procedures.

### *3.6.2 Compression Queues*

Thread-specific generic batch queues control codebook compressions. These queues drive other batch queues (called *codebook queues*) that perform the actual compression work. System load dictates how processing is distributed among the queues. Figure 14 shows a generic batch queue and its associated codebook queues.

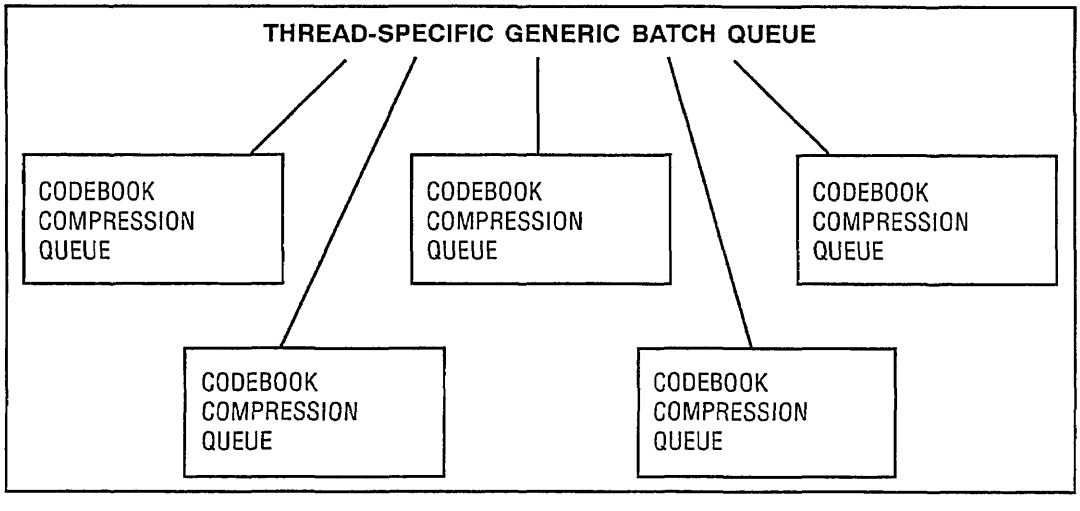



Fig. 14 — Thread-specific generic batch queue and associated codebook compression queues

The Sqcode utility lists current generic batch and codebook compression queue assignments. Figure 15 provides a sample Sqcode listing, including the generic batch queues for processing threads A2A (A2A_CODEBOOKS) and A2C (A2C_CODEBOOKS), and associated codebook compression queues for A2A (CODEBOOK1 and CODEBOOK2). Job file names in compression queues use the following convention:

xxx_Cmmmmmssss,

where

xxx is the processing thread name,
C indicates PASS1 compression, a "P" is used to indicate PASS3 compression,
mmmmm is a five-digit MDFF sequence number, and
ssss is a four-digit codebook status file entry.

Thread-specific queue assignments may need to be changed periodically to keep processing loads balanced. To meet this requirement, the SET_GENERIC_CB_QUEUE option of Chart_Control provides the capability of dynamically modifying the generic codebook queue for the current processing thread. In addition, some of the more common queue manipulation commands are provided in Fig. 16. Thread-specific queues are automatically stopped whenever the CPS is booted. This allows the CPS to come to a known state after booting.

**Generic Batch Queue A2A_CODEBOOKS**

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 458 | A2A_C012340013 | CAC | Pending |
| 459 | A2A_C012340014 | CAC | Pending |
| 460 | A2A_C012340015 | CAC | Pending |
| 461 | A2A_C012340016 | CAC | Pending |
| 462 | A2A_C012340017 | CAC | Pending |
| 463 | A2A_C012340018 | CAC | Pending |
| 464 | A2A_C012340019 | CAC | Pending |
| 465 | A2A_C012340020 | CAC | Pending |

**Generic Batch Queue A2C_CODEBOOKS**

**Batch Queue CODEBOOK1, Busy, on ZOMBIE::**

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 446 | A2A_C012340001 | CAC | Executing |
| 447 | A2A_C012340002 | CAC | Executing |
| 448 | A2A_C012340003 | CAC | Executing |
| 454 | A2A_C012340009 | CAC | Executing |
| 455 | A2A_C012340010 | CAC | Executing |
| 456 | A2A_C012340011 | CAC | Executing |

**Batch Queue CODEBOOK2, Busy, on ZOMBIE::**

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 449 | A2A_C012340004 | CAC | Executing |
| 450 | A2A_C012340005 | CAC | Executing |
| 451 | A2A_C012340006 | CAC | Executing |
| 452 | A2A_C012340007 | CAC | Executing |
| 453 | A2A_C012340008 | CAC | Executing |
| 457 | A2A_C012340012 | CAC | Executing |

Fig. 15 — Sample Sqcode listing. Note that processing thread names are incorporated into job names.

| Command | | Action |
|---|---|---|
| *stop/queue/next* | **queue name*** | Stops a queue after the last job in the queue terminates |
| *start/queue* | **queue name*** | Starts a queue |
| *stop/queue/reset* | **queue name*** | Stops a queue without allowing the last job to complete. Currently executing jobs are terminated. |

*where **queue name** is the name of the queue to be manipulated

Fig. 16 — Common queue manipulation commands

Batch Queue MDFF_BATCH, Available, on ZOMBIE::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 897 | MDFF_CONTROL | SYSTEM | Executing |
| 351 | A2A_PROCESSCHRT | CAC | Executing |
| 534 | A2C_PROCESSCHRT | CAC | Executing |

Fig. 17 — Sample Sqmdff output listing

Jobs already submitted to a particular queue typically remain there until they are completed. However, submitted jobs may be sent to another queue (i.e., requeued) through the command:

*stop/queue/requeue/entry* = **entry_number QUEUE_NAME.**

For example, the following command requeues entry 758 from its current queue to the MDFF_BATCH queue:

*stop/queue/requeue/entry* = **758 MDFF_BATCH.**

MDFF_BATCH queue activity is monitored by the Sqmdff utility:

*sqmdff* (no parameters).

Normally, every active processing thread should have a Process_Chart job with an EXECUTING status in MDFF_BATCH (Fig. 17).

If a processing thread's Process_Chart job does not appear in the MDFF_BATCH queue, it should be resubmitted to the queue. Prior to its resubmission, examine the contents of Process_Chart's log file (xxx_PROCESSCHRT.LOG, where xxx is the processing thread name) that is stored in the MDFF_SCRATCH directory, and check for error messages since the system traps error conditions and records them in this file. If the error condition is not obvious or is unclear, report the problem to the CPS software maintenance group.

Resubmit Process_Chart by executing Chart_Control, as follows:

- Assign processing thread xxx by typing:

  *thread* **xxx** (**xxx** is the processing thread name).

- Execute Chart_Control by typing:

  *chart_control.*

Chart_Control *automatically* verifies that the MDFF_BATCH queue contains a Process_Chart job for the current processing thread. If a job does not exist, Chart_Control will submit one. Use the/NOSUBMIT qualifier to prohibit automatic submission of Process_Chart to the MDFF_BATCH queue.

## 3.7 Disk Storage

Logical device names are used to refer to CAC processing storage devices (e.g., disks) and directories since they are more descriptive than their physical device name counterparts. Figure 18 lists the major CPS disk drives and directories that have significant storage requirements.

Information about storage devices and disk space utilized by a specific processing thread is available by typing the command *diskuse*. Figure 19 provides a sample listing from the Diskuse utility. The system manager should be consulted when disk space becomes low.

## 3.8 File Backup

As processing progresses, backups of each thread must be created to enable recovery from system failure or other loss of data. A backup of the CPS should also be created if any permanent changes are made to the CPS software and system configuration. This section describes backup procedures for processing threads and for the CPS.

### *3.8.1 Processing Thread Backup*

Chart_Control provides a backup option to save an entire processing thread's files. This option is available in almost every Chart_Control menu. Generally, it is best to backup a processing thread frequently in PASS1 (every 8 to 10 ADRG CDROMs processed) and again before starting each major processing phase. This provides a clean starting point should a processing thread need to be restored. As the user becomes more confident, the number of backups performed may be reduced.

| LOGICAL NAME | DESCRIPTION |
|---|---|
| CHART_SEGS | A storage device for all processing threads. Used to build other logical names. |
| CHART_SEGMENTS | A directory path for the currently defined processing thread.<br>CHART_SEGS is included as part of this path. Accommodates downsampled data. |
| CHART_ODI | The name of a storage device that accommodates compressed data for all processing threads. |
| CHART_ODI_DISK | A directory specification (which includes CHART_ODI).<br>Part of the full path for the currently defined processing thread. |
| MDFF_SCRATCH | A directory path for the currently defined processing thread.<br>Accommodates compression command files, log files, and miscellaneous files. |
| ISO_IMAGES | A directory path for all processing threads.<br>Accommodates ISO 9660 related files, including command and log files, and the final ISO image file. |
| ADRGDIRxx:[FILES] | A directory that stores copied ADRG data. xx is a number from 01 through 06 that corresponds to the CDROM reader number. |

Fig. 18 — Major CPS storage devices and directories

| Device | Volume Label | Used Blocks | Free Blocks | Total Blocks |
|---|---|---|---|---|
| ZOMBIE$DKA0 | AXPVMSSYS | 2163932 (52%) | 1946548 (48%) | 4110480 |
| ZOMBIE$DKA100 | MDFF_SYSTEM | 784896 (19%) | 3325584 (81%) | 4110480 |
| ZOMBIE$DKA200 | MDFF_SCRATCH | 89892 (2%) | 4020588 (98%) | 4110480 |
| ZOMBIE$DKB0 | CHARTSEGS_1 | 17040 (0%) | 4093440 (100%) | 4110480 |
| ZOMBIE$DKB100 | CHARTODI_1 | 82800 (2%) | 4027680 (98%) | 4110480 |
| ZOMBIE$DKB200 | WORK1 | 7980 (0%) | 4102500 (100%) | 4110480 |
| ZOMBIE$DKC0 | CHARTSEGS_2 | 17040 (0%) | 4093440 (100%) | 4110480 |
| ZOMBIE$DKC100 | CHARTODI_2 | 82224 (2%) | 4028256 (98%) | 4110480 |
| ZOMBIE$DKC200 | WORK2 | 6980 (0%) | 4103500 (100%) | 4110480 |
| ZOMBIE$DKD100 | | Not mounted | | 6980 |
| ZOMBIE$DKD200 | | Not mounted | | 6980 |
| ZOMBIE$DKD400 | | Not mounted | | 6980 |
| ZOMBIE$DKD500 | | Not mounted | | 6980 |
| ZOMBIE$DKE0 | | Not mounted | | 6980 |
| ZOMBIE$DKE300 | | Not mounted | | 6980 |
| Totals on 15 structures: | | 3294664 (**%) | 33741536 (**%) | 36994320 |

Fig. 19 — Diskuse Command Listing

To backup a processing thread, run Chart_Control and select the BACKUP_ODI menu option. Chart_Control prompts for the tape device to use for the thread backup. Enter **8 mm** to write to the 8-mm tape drive or **DLT** to write to the DLT tape drive. A batch job is then submitted to the MDFF_BATCH queue.

Only the thread-specific files in the CHART_SEGMENTS, CHART_ODI_DISK, and MDFF_SCRATCH directories are backed up. The files located in the ADRGDIR## (where ## denotes CDROM reader number) directories are ***not*** backed up since these are working scratch areas and are easily restored form their original ADRG CDROMs.

All CAC users are sent a mail message when the backup procedure completes. Note, only one backup procedure should be submitted at a time. No check is made to ensure this. The backup log file should be examined to verify that the backup operation completed normally (via the VMS TYPE command or the editor):

MDFF_SYSTEM:[BACKUP]BACKUP_###_FILES.LOG, where ### is the processing thread name.

The tape cartridge should be removed from the tape drive and the write-lock switch should be set in the "lock" position. Tape labels typically include the date of the backup and the text "BACKUP_###_FILES," where ### denotes the processing thread name. Figure 20 provides an example of a backup tape label for thread A2A recorded on September 5, 1995.

After the ISO image has been built, a final backup of the thread must be created. This tape's label also includes the words "FINAL BACKUP" (Fig. 21).

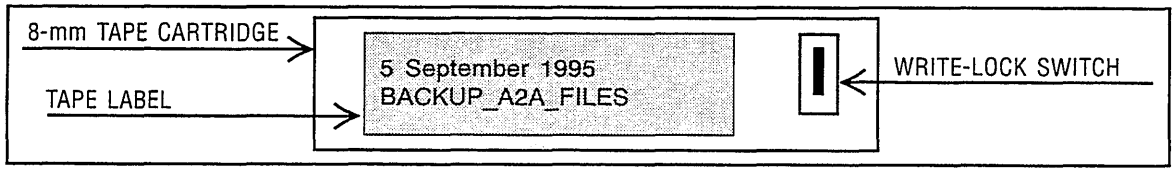



Fig. 20 — Backup tape label example

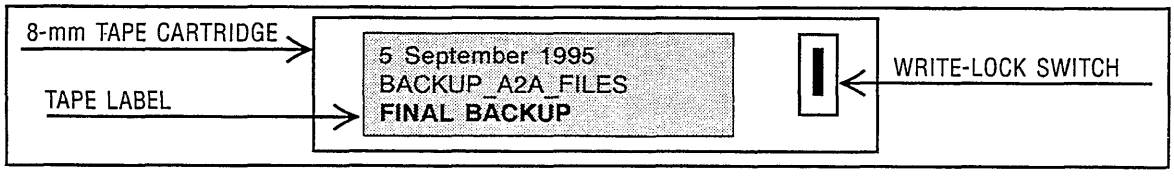



Fig. 21 — Final backup tape label for an ISO image

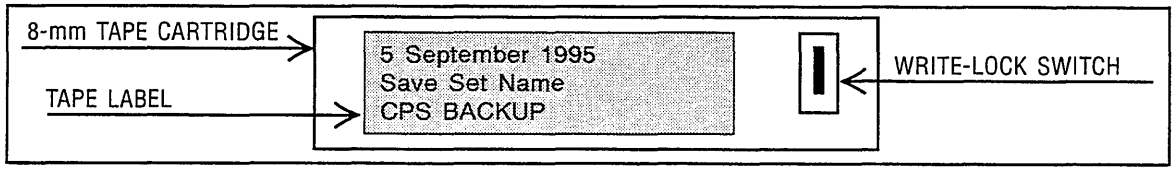



Fig. 22 — Tape label for the CPS backup

### *3.8.2 CPS Backup*

A backup of the CPS should be created if any permanent changes are made to the CPS software and system configuration. Use the following procedure to backup the CPS:

@MDFF_SYSTEM:[SOFTWARE.COM]Submit_backup_MDFF_system.

This tape's label includes the words "CPS BACKUP" (Fig. 22).

## 4.0 CAC PROCESSING INITIALIZATION

### 4.1 Introduction

A processing thread must be initialized before CAC processing can begin. As described in Secs. 2.3 and 3.1, a processing thread name is required for each CAC creation since it identifies all the files and directories associated with a particular CAC ODI build. The INITIALIZATION phase creates a new processing thread and all required files and directories that are used in a CAC ODI build.

There are two types of INITIALIZATION: (1) to build a new CAC ODI and (2) to update an existing CAC ODI. Updates of existing CACs are required when new data for the area of coverage becomes available, including new charts of previously uncovered areas, as well as revised charts of previously processed areas.

INITIALIZATION is similar for new and updated CAC ODIs (i.e., most of the prompts are the same), but there are some differences that are discussed in the following sections.

### 4.2 Common Beginning Protocol

Prior to invoking Chart_Control, the user should deactivate any preexisting processing thread with the following command:

*thread* (no parameter).

The user can then invoke Chart_Control, which displays the INITIALIZATION menu, allowing the user to define a new processing thread (Fig. 23).

Note that if no processing thread is defined, "N/A" (i.e., Not Available) is shown after "Processing Code:." At the bottom of the menu the status window message states that no processing thread is selected and the default value NO_OP is used for the *current processing phase* and *last completed phase* flags at the top of the menu.

If the user chooses the default response ("NO") to the prompt in the center of the menu, no new processing thread is created. If the user enters "YES" (upper or lower case), INITIALIZATION continues. The same prompt is repeated as a double check to avoid the unintentional creation of a processing thread.

Next, Chart_Control prompts the user for the processing thread name (the three-character format described in Sec. 3.1) as shown in Fig. 24.

Chart_Control then prompts the user for the codebook queue name(s) (Fig. 25). The user can specify multiple queues by separating their names with commas (e.g., codebook1, codebook2).

After the user has entered all the desired queue names, Chart_Control displays an editable list of all the queues specific to this processing thread. The user should examine the list of queue names, correct any errors, and press ^Z to exit and permanently save these changes. The Chart_Control menu then returns to the screen.

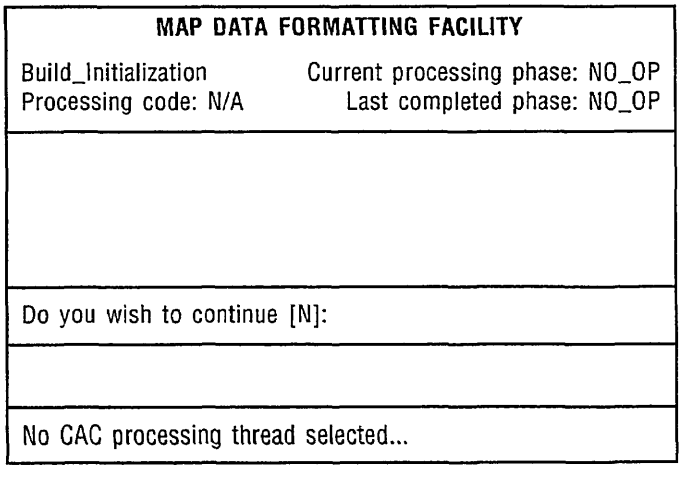



Fig. 23 — First INITIALIZATION menu prompt

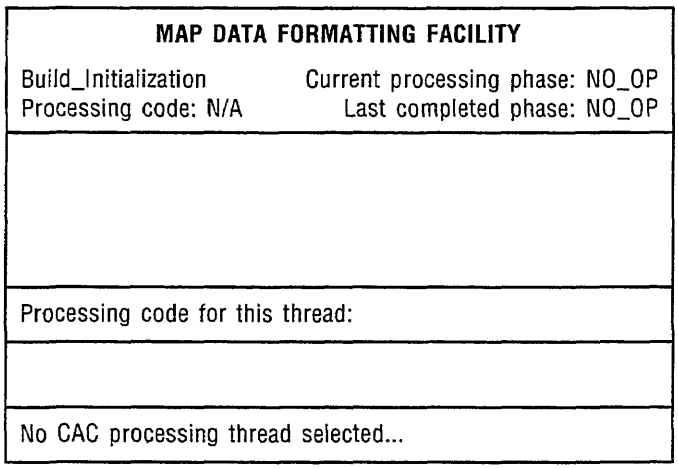



Fig. 24 — Processing thread name prompt

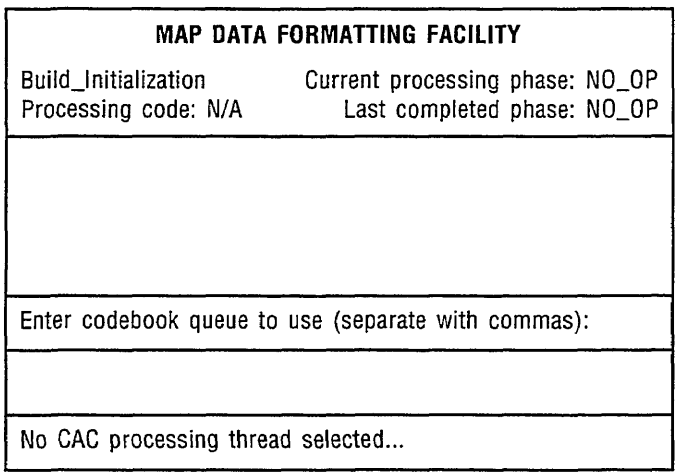



Fig. 25 — Prompt for CODEBOOK queue names

Chart_Control next prompts the user to set the new processing thread for use (Fig. 26). The default response is "YES."

Once Chart_Control has defined all necessary logical names, the menu issues a status message that miscellaneous procedures (e.g., backup command procedures) and files (e.g., CHART_STATUS. DAT) are being built.

Chart_Control then asks the user if this is an updated CAC (Fig. 27). The default is *not* an UPDATE (see Sec. 4.4 for UPDATE procedures). Hence, a simple carriage returns a new CAC ODI build. Note that the newly created processing thread name now appears in the menu.

## 4.3 Initialization for a New CAC

Chart_Control next prompts the user for the chart scale (Fig. 28). All possible chart scales are displayed. Note the message indicating this will *not* be an UPDATE (i.e., this will be a new CAC).

Use the arrow keys to highlight the appropriate scale (e.g., TPC in Fig. 28), then press the return key. The message window will indicate the selected scale (Fig. 29).

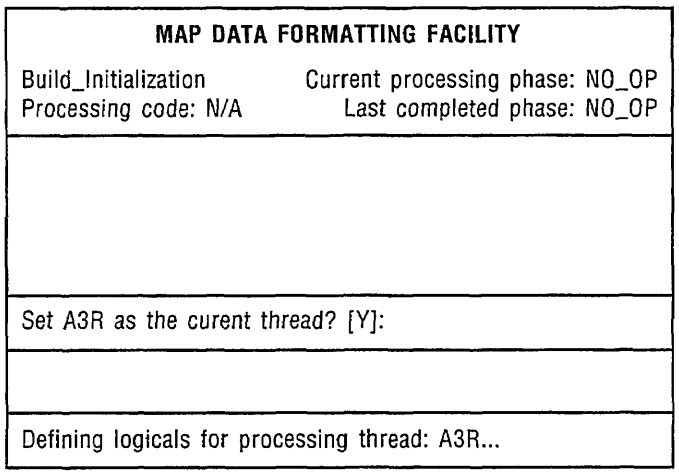



Fig. 26 — Processing thread and logical name definitions status message. The status message is displayed after the processing thread has been made current.

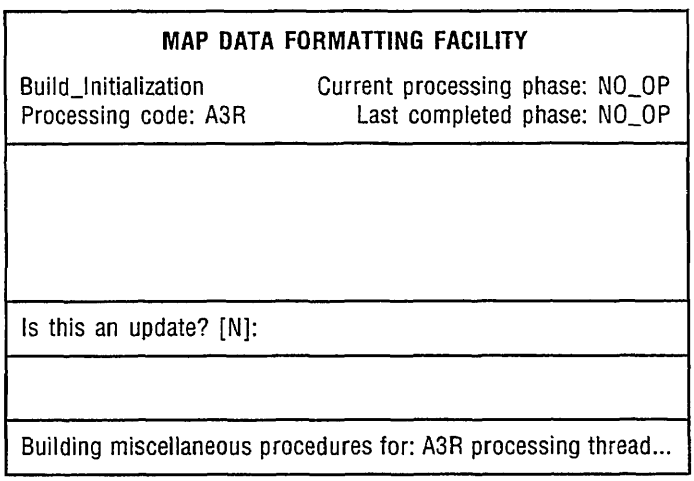



Fig. 27 — UPDATE prompt

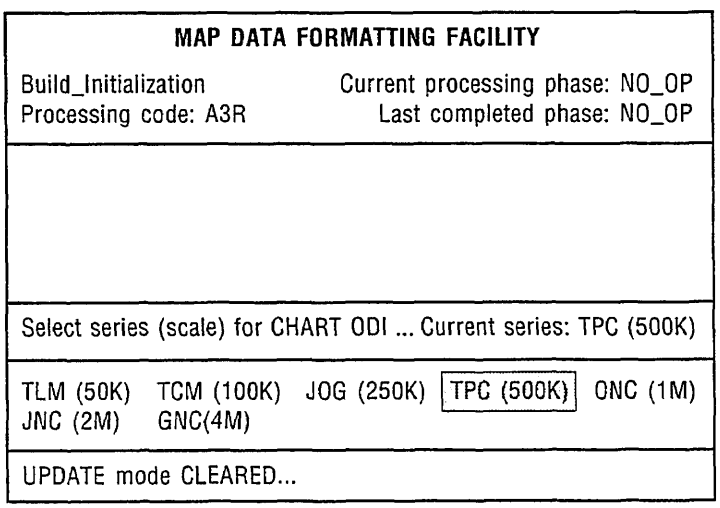



Fig. 28 — Chart scale selection prompt

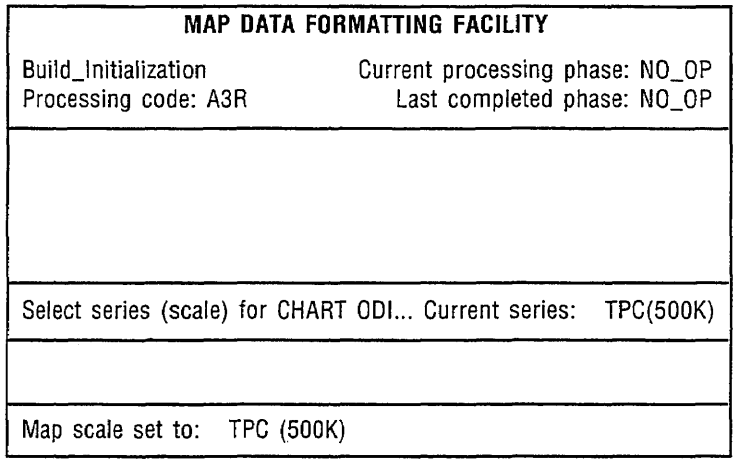
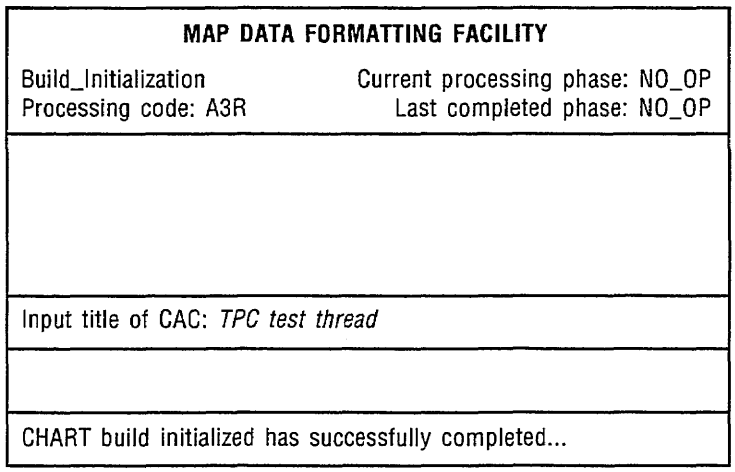



Fig. 29 — Chart scale selection message




*< This text will become the CAC logbook cover sheet title*

Fig. 30 — INITIALIZATION completion status message

Finally, Chart_Control prompts the user for a title for the CAC Log Book Cover Sheet. After the user enters a title, all required INITIALIZATION information will have been obtained, as indicated by the status message (Fig. 30).

Chart_Control now advances to the PASS1 phase described in Sec. 5.0. The completion of this INITIALIZATION phase is depicted in the status message.

### 4.4 Initialization for a CAC Update

CAC ODI UPDATES are required when new data for the area of coverage becomes available, including new charts of previously unprocessed areas or revised charts (of previously processed areas).

INITIALIZATION for a CAC ODI UPDATE begins when the user replies "Y" to the prompt for an UPDATE (Fig. 27). Chart_Control lists all archived CAC ODIs (Fig. 31). The user scrolls through the list and selects the CAC ODI to be updated (the line that is **highlighted** will be selected).

```
                    MAP DATA FORMATTING FACILITY

Build_Initialization                  Current processing phase: NO_OP
Processing code: A3R                    Last completed phase: NO_OP

1 MDFF_SYSTEM:[ARCHIVE]0001B_CAC_ODI_STATUS.DAT
2 MDFF_SYSTEM:[ARCHIVE]0001C_CAC_ODI_STATUS.DAT
3 MDFF_SYSTEM:[ARCHIVE]0002A_CAC_ODI_STATUS.DAT
4 MDFF_SYSTEM:[ARCHIVE]0002B_CAC_ODI_STATUS.DAT
5 MDFF_SYSTEM:[ARCHIVE]0003A_CAC_ODI_STATUS.DAT
6 MDFF_SYSTEM:[ARCHIVE]0004A_CAC_ODI_STATUS.DAT
7 MDFF_SYSTEM:[ARCHIVE]0005A_CAC_ODI_STATUS.DAT

Use UP/DOWN arrow keys to scroll, hit <CR> to make selection.

Library update mode set !!!
```

Fig. 31 — Archived CAC ODIs listing

The first four characters of the archived file name represent the CAC ODI number; the fifth character is the version number or edition. For example, the highlighted list entry in Fig. 31 is "**2 MDFF_SYSTEM:[ARCHIVE]0001C_CAC_ODI_STATUS.DAT**," which corresponds to CAC number 0001, version "C." If selected, the UPDATE to this CAC ODI would be number 0001, version "D."

Next, Chart_Control prompts the user for the CDROM reader containing the CAC CDROM to be updated (Fig. 32). Prior to responding, the user must place that CAC CDROM in the designated CDROM reader; after responding to the prompt, Chart_Control will verify that the CDROM loaded in the specified reader matches the CAC selected for UPDATE. If they do not match, the user must check his or her selections and correct the error (i.e., ensure that he correct CAC ODI file name is selected and that the corresponding CAC CDROM is loaded in the specified reader.

Following successful verification, Chart_Control builds and submits a command procedure to copy all necessary files from the CAC CDROM (once INITIALIZATION is complete, use the

```
                    MAP DATA FORMATTING FACILITY

Validate_Input_Media                  Current processing phase: NO_OP
Processing code: A3R                    Last completed phase: NO_OP

Select an input device for validation:

CDROM01   CDROM02   CDROM03   CDROM04   CDROM05   CDROM06
```

Fig. 32 — UPDATE CAC CDROM reader selection

Sqmdff utility to check the MDFF_BATCH queue and verify that the copy is occurring; it will exist as the process "RELOAD_CAC").

Finally, Chart_Control prompts the user for a title to the CAC Log Book Cover Sheet. After the user enters a title, all required INITIALIZATION information will have been obtained, as indicated by the status message (Fig. 33).

Control advances to an intermediate WAIT_STATE (Fig. 34) until the CAC ODI has been copied from CDROM and the batch process for "RELOAD_CAC" has completed in the MDFF_BATCH queue. After RELOAD_CAC has completed, control will automatically advance to the PASS1 Phase (see Sec. 5.0).

### 4.5 Error Recovery

Typical errors that occur during INITIALIZATION include system failure (i.e., power loss) and insufficient disk space during the update reload operation (i.e., the disk becomes full). Error recovery consists of deleting the processing thread (Sec. 3.3.1), correcting the cause of the error, then repeating INITIALIZATION.

| **MAP DATA FORMATTING FACILITY** | |
|---|---|
| Validate_Input_Media<br>Processing code: A3R | Current processing phase: NO_OP<br>Last completed phase: NO_OP |
| | |
| | |
| Input title of CAC: *UPDATE of 1C* | |
| CHART build initialized has successfully completed... | |

Fig. 33 — INITIALIZATION completion message for a CAC UPDATE

| **MAP DATA FORMATTING FACILITY** | |
|---|---|
| Wait_State<br>Processing code: A3R | Current processing phase: INIT_RELOAD_CDROM_DATA<br>Last completed phase: INIT_THREAD |
| | |
| | |
| Select a function... | |
| EXIT UPDATE_SCREEN BACKUP_ODI (Tue Jan 2 00:00:00 1996) | |
| The current processing phase is an intermediate step.<br>Once it has completed, an appropriate menu will be provided. | |

Fig. 34 — WAIT_STATE status message for CAC UPDATE

## 5.0 PASS1 PROCESSING

### 5.1 Introduction

PASS1 is comprised of four events: reading ADRG data, downsampling (converting from ARC to TS), color compression, and spatial compression of core segments. As shown in Fig. 2, core segments cover the innermost portions of a chart. An overview of PASS1 functionality is shown in Fig. 35.

During PASS1, the current *processing_step* field in the CHART_STATUS.DAT header record contains the value PASS1_PROCESSING. The PASS1 main menu provides options to validate and select CDROM readers, backup all files associated with this processing thread, manage queues, and manipulate processing control (Fig. 36).

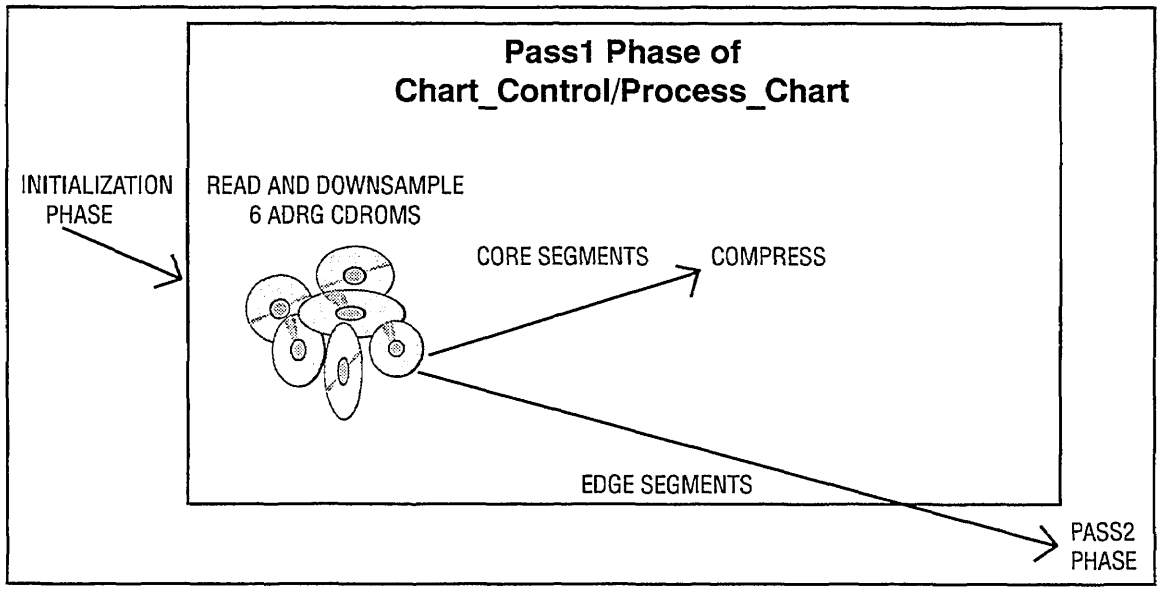



Fig. 35 — PASS1 overview

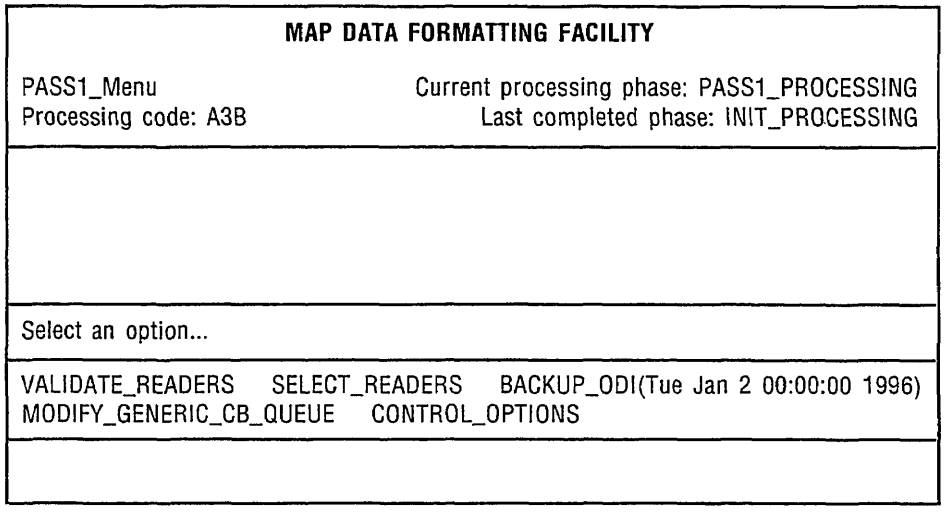



Fig. 36 — The PASS1 main menu

The VALIDATE_READERS and SELECT_READERS menu options are used to read ADRG data. Subsequent PASS1 processing (i.e., downsampling and data compressions) occur automatically with no user interaction. The remaining menu options are used to monitor processing (e.g., queue management). The following sections describe the main PASS1 events and the menu options in more detail.

## 5.2 Copying ADRG Data from CDROM

### *5.2.1 Use of Chart_Control*

The user may select up to six ADRG CDROMs to copy at one time since the CPS is equipped with six CDROM readers. Load the first six ADRG CDROMs on these readers, then specify the appropriate processing thread (e.g., thread A3B) and execute Chart_Control (if not already running). Make the following selections from the PASS1 menu:

- Select VALIDATE_READERS (Fig. 37) to list which CDROMs are loaded on each available reader.
- Scroll through the list of CDROMs (Fig. 38) to **highlight** the reader containing the first ADRG CDROM to be validated and press *RETURN*. Information pertaining to the selected CDROM will be displayed in the reader selection window. Chart_Control will search the CPS's ADRG database file (CDROM_INFO.DAT) for the NIMA stock number and edition of the selected CDROM. If a match occurs, an MDFF sequence number is returned. When no match occurs, Chart_Control returns a value of zero. Every ADRG CDROM that is processed by the CPS has its own unique MDFF sequence number. An MDFF sequence number of zero indicates that the ADRG CDROM has never been copied by the CPS. After it is copied, it will be automatically assigned a new MDFF sequence number and logged in the chart database.
- Enter ^Z to return to the main PASS1 menu. The PASS1 menu now reflects validation information, including the NIMA stock numbers and associated MDFF sequence numbers of the selected ADRG CDROMs (Fig. 39).
- Choose the *SELECT_READERS* option to select the CDROM reader that will access the validated ADRG CDROM.

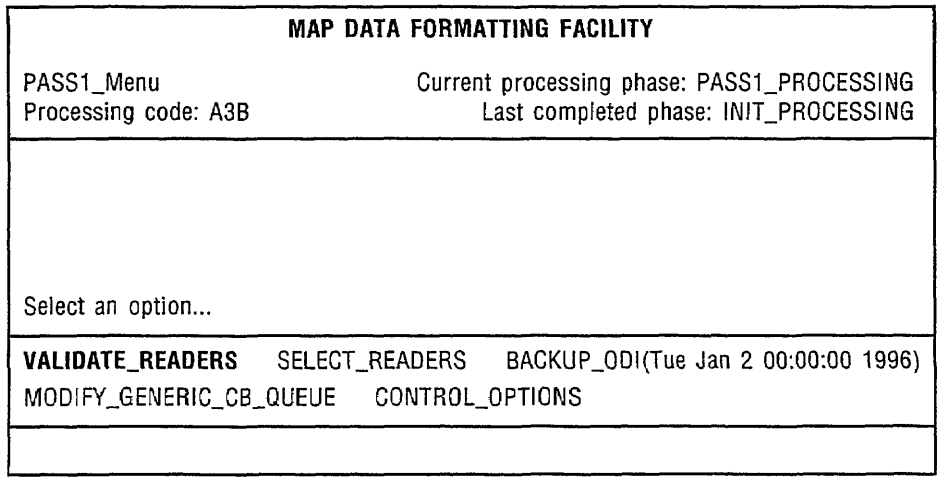



Fig. 37 — Selection of the VALIDATE_READERS option from the PASS1 menu

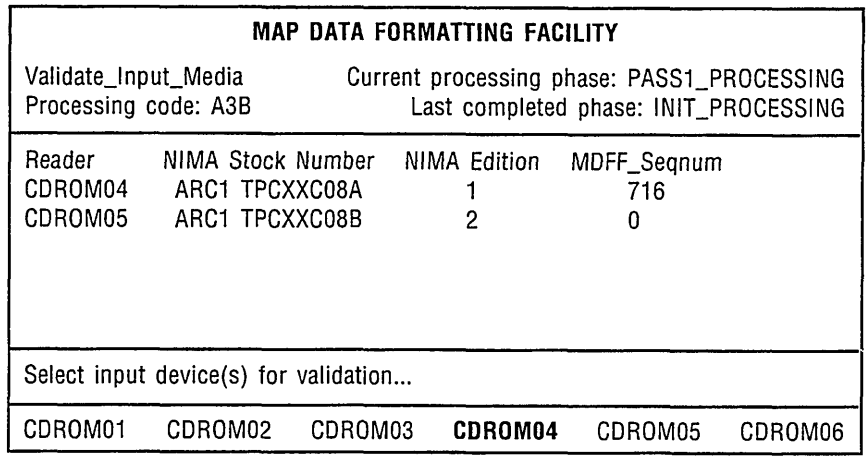



Fig. 38 — ADRG CDROM validation from VALIDATE_READERS menu

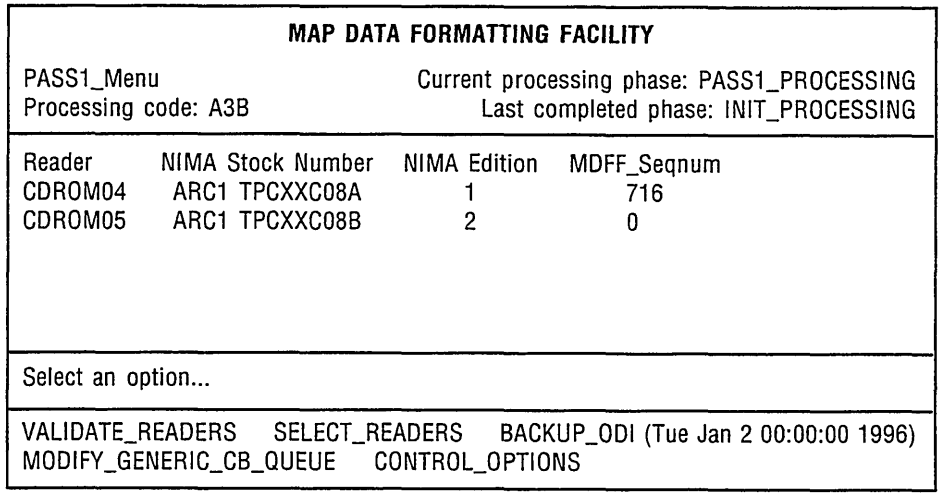



Fig. 39 — Post validation view of PASS1 main menu

- **Highlight** the reader containing the ADRG CDROM to be copied (Fig. 40) and press *RETURN*. Repeat this step for each ADRG CDROM to be selected.
- Enter ^Z to return to the main PASS1 menu.
- Enter ^Z to exit Chart_Control.

*IMPORTANT NOTE:* The ADRG CDROM *must remain* in the reader until Chart_Control authorizes its removal (via mail message). This ensures that all processing tasks requiring its use are complete.

### *5.2.2 The CHART_STATUS.DAT File and Storage Requirements*

Once all required ADRG CDROMs have been validated and selected for processing, the List_Chart_Status utility can verify the successful addition of these CDROMs to the CHART_STATUS. DAT file. Initially, these new entries will have a status of NO_OP for the status_record fields *processing_step* and *last_completed_step* (Table 1).

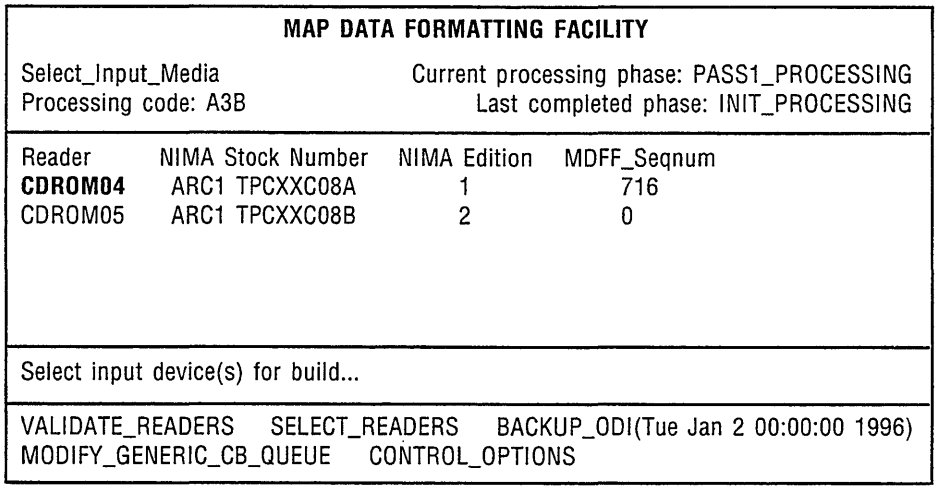



Fig. 40 — ADRG CDROM selection from the SELECT_READERS menu

Table 1 — Values of Status Record Control Fields ("Processing_Step" and "Last_Completed_Step") in CHART_STATUS.DAT File During PASS1 Processing. Values are Provided for Each ADRG CDROM Being Processed for a Given Thread.

| DESCRIPTION OF PASS1 PROCESSING STEPS | PROCESSING_STEP | LAST_COMPLETED_STEP |
|---|---|---|
| **Copying ADRG Data from CDROM (Sec. 5.2)** | | |
| Initial values, immediately after validation and selection (Sec. 5.2.2) | NO_OP | NO_OP |
| While copying ADRG CDROM (Sec. 5.2.2) | COPY_ADRG | NO_OP |
| After ADRG CDROM successfully copied (Sec. 5.2.2) | NO_OP | COPY_ADRG |
| **Downsampling from ARC to TS (Sec. 5.3)** | | |
| During downsampling (Sec. 5.3.1) | PASS1_DOWNSAMPLE | COPY_ADRG |
| Immediately after downsampling (Sec. 5.3.2) | NO_OP | PASS1_DOWNSAMPLE |
| **Compressing Core Segments (Sec. 5.4)** | | |
| Building PASS1 compression procedures (Sec. 5.4.1) | BUILD_ADRG_CD_PROCS | PASS1_DOWNSAMPLE |
| Submitting codebook compression procedures (Sec. 5.4.1) | SUBMIT_ADRG_CD_PROCS | BUILD_ADRG_CD_PROCS |
| Building compression codebooks (Sec. 5.4.1) | BUILD_ADRG_CBS | SUBMIT_ADRG_CD_PROCS |

ADRG data are copied onto disk with the logical name ADRGDIR##, where ## is the CDROM reader number. Process_Chart verifies that sufficient space is available on the disk, then copies the ADRG CDROMs. If there is insufficient disk space, Process_Chart will continually recheck until adequate space becomes available, then it will copy the ADRG CDROM.

Once copying begins, Chart_Control updates the CHART_STATUS.DAT status_record processing_step field as COPY_ADRG and last_completed_step field as NO_OP (Table 1).

Execute the sqmdff utility to monitor ADRG copies by listing the jobs in the MDFF_BATCH queue named ###_COPYADRG**, where ### is the processing thread name and ** is the CDROM reader number.

After each ADRG CDROM has been successfully copied, Chart_Control again updates the CHART_STATUS status_record processing_step field as NO_OP and last_completed_step field as COPY_ADRG (Table 1).

### *5.2.3 Validation and Selection Errors: Detection and Handling*

There are several errors that can occur during ADRG CDROM validation and selection. This section describes these errors and how to correct them.

#### 5.2.3.1 Incorrect Scale

This error occurs during validation when an ADRG CDROM's scale does not match the scale of the current processing thread. This usually occurs when multiple processing threads at different scales are being used concurrently. For example, Fig. 41 shows the message that is displayed when a Joint Operational Graphics (JOG) ADRG CDROM is loaded on a CD-READER that was selected for a TPC processing thread.

When working with concurrent processing threads, pay close attention when validating and selecting ADRG CDROMs. Chart_Control cannot detect an error if a CDROM at the *same scale* as the currently assigned thread *was meant to be selected by another thread* at the same scale.

**MAP DATA FORMATTING FACILITY**

Validate_Input_Media — Current processing phase: PASS1_PROCESSING
Processing code: A3B — Last completed phase: INIT_PROCESSING

Reader | NIMA Stock Number | NIMA Edition | MDFF_Seqnum

Select input device(s) for validation...

CDROM01 | CDROM02 | CDROM03 | CDROM04 | CDROM05 | CDROM06

Scale of ARC1 1501A0EUD does NOT match the current thread's map scale.
Validation ABORTED... check CDROM...

Fig. 41 — ADRG CDROM and processing thread scale mismatch error message

#### 5.2.3.2 Validation of a Previously Processed ADRG CDROM

This error occurs when a previously processed ADRG CDROM is selected for validation. Note that Chart_Control *requires a user response* since ADRG CDROMs occasionally need to be reprocessed (Fig. 42).

Enter either "Y" to reprocess this CDROM or "N" (default) to stop the validation. When "Y" is entered, the control flags in CHART_STATUS.DAT are set to reprocess this CDROM. The SELECT_READERS option will not be performed, since this CDROM is automatically selected for processing.

### 5.3 Downsampling

*5.3.1 CHART_STATUS.DAT Values*

After all required ADRG data are successfully copied, Process_Chart begins downsampling to transform the data from ARC to TS projection and compress it by a factor of 4:1 [6]. The downsampled chart data are stored as TS segments, where each segment is 256 × 256 pixels and represents a 2 × 2 in. area of paper chart.

Downsampled segments are written onto a disk with the logical name CHART_SEGS. A downsampled TS segment is comprised of three files, one file for each color component: red, green, and blue (RGB). These files utilize the file naming convention shown in Fig. 43.

| MAP DATA FORMATTING FACILITY | |
|---|---|
| Validate_Input_Media<br>Processing code: A3B | Current processing phase: PASS1_PROCESSING<br>Last completed phase: INIT_PROCESSING |
| Reader NIMA Stock Number NIMA Edition MDFF_Seqnum | |
| Reprocess ARC1 TPCXXL26A from scratch [N]: | |
| CDROM01 CDROM02 CDROM03 CDROM04 CDROM05 CDROM06 | |
| *WARNING* ARC1 TPCXXL26A is ALREADY part of current CHART ODI build *WARNING*! | |

Fig. 42 — Previously processed ADRG CDROM error message

Downsampled TS Segment file name is xxxxxxxxxx.y_zz where

xxxxxxxxxx = a 10-digit file name computed with the formula 18,001×(column+9000)+row+9001
y = one letter representing the RGB component (either R, G, or B)
zz = a two-letter TS zone code (SP, ST, EQ, NT, or NP)

Fig. 43 — Downsampled TS segment file name formula

For example, a TS segment in the equatorial region (EQ) with a row number of 10 and a column number of 15 would be comprised of the following files:

| | |
|---|---|
| 0162288026.R_EQ | contains the red image components |
| 0162288026.G_EQ | contains the green image components |
| 0162288026.B_EQ | contains the blue image components |

During downsampling, Chart_Control updates CHART_STATUS.DAT status_record processing_step field as PASS1_DOWNSAMPLE and last_completed_step field as COPY_ADRG (Table 1).

Execute the Sqmdff utility to display the MDFF_BATCH queue, which will contain the job ###_DOWNSAMP**, where ### is the processing thread name and ** is the CDROM reader number.

### *5.3.2 Downsample Errors: Detection and Handling*

When a downsample job finishes, Chart_Control again updates the CHART_STATUS status_record processing_step field as NO_OP, and last_completed_step field as PASS1_DOWNSAMPLE (Table 1). Execute List_Chart_Status, Sqmdff, and Sqcode to monitor the progress of queued batch jobs.

#### 5.3.2.1 Disk Storage Related Errors

When a CHART_STATUS.DAT entry is marked as still downsampling, but Sqmdff does *not list it in the batch queue*, an error has occurred. There are two probable causes for downsampling errors. The first type occurs during execution of the downsample batch job, either due to a lack of disk storage or attempting to write to a bad disk sector. Examine the batch job log file to determine the cause of the error and then the corrective action. Batch job log files are stored in the MDFF_SCRATCH:[000000] directory as ###_DOWNSAMPL**.LOG, where ### is the processing thread name and ** is the CDROM reader number.

Execute the Diskuse utility to monitor disk space availability. Each downsampled ADRG CDROM requires a minimum of 100 Mbytes on the CHART_SEGS disk(s). If there is insufficient space, wait for the CDROMs in the compression phase to complete, since downsampled data will then be deleted from CHART_SEGS. Once sufficient disk space becomes available on CHART_SEGS, start the next job.

The number of downsample jobs that can concurrently execute within the batch queue is controlled with the logical "CONCURRENT_CDS." To check the current value of CONCURRENT_CDS, type the following command:

*$show logical* **CONCURRENT_CDS.**

The output will look like this:

"CONCURRENT_CDS" = "3" (A3B),

where "3" is the total number of ADRG copy and downsample jobs that are allowed to run (in batch processing) at one time per processing thread and (A3B) is the currently defined processing thread name.

This value should be changed based on disk space availability. For example, when there is only one processing thread, this value is usually increased to maximize processing. However, when there are several processing threads in use, this value should be decreased to limit each processing thread's storage consumption. Type the following command to change the number of downsample jobs allowed to execute concurrently:

$$\$ \textit{define/table} = \textbf{A3B CONCURRENT_CDS 2.}$$

The system will respond with the following information:

%I-SUPERSEDE, previous value of CONCURRENT_CDS has been superseded.

Now two batch processing jobs can execute concurrently. The redefinition only takes effect for the processing thread A3B. Note that 3 is the *default value,* and if the system is rebooted, the value of CONCURRENT_CDS will be set back to 3.

#### 5.3.2.2 Power Loss Related Errors

Another type of processing error occurs when the system loses power during processing. Use the RECOVER_CHART_BUILD option in Chart_Control to recover and resume processing (Fig. 44).

After selecting the RECOVER_CHART_BUILD option, Chart_Control prompts the user to proceed with the recover build operation. The default is "No."

Enter "Y" to continue the recover build operation and display the current processing step information *for each entry* in CHART_STATUS.DAT. As each entry is displayed, indicate whether processing should be restarted by entering "Y." See Sec. 5.7.3.1 for additional information regarding RECOVER_CHART_BUILD.

### 5.4 Color and Spatial Compressions

Compression is achieved through vector quantization (VQ). VQ compression generates a *codebook* as part of this compression technique. In the prototype CAC processing system, VQ generated one codebook to compress 25 segments. At that time, emphasis was placed on the correct generation of codebooks because one codebook was used for many segments. However, in the current CAC processing system, one codebook is generated to compress only one segment. Hence, emphasis has

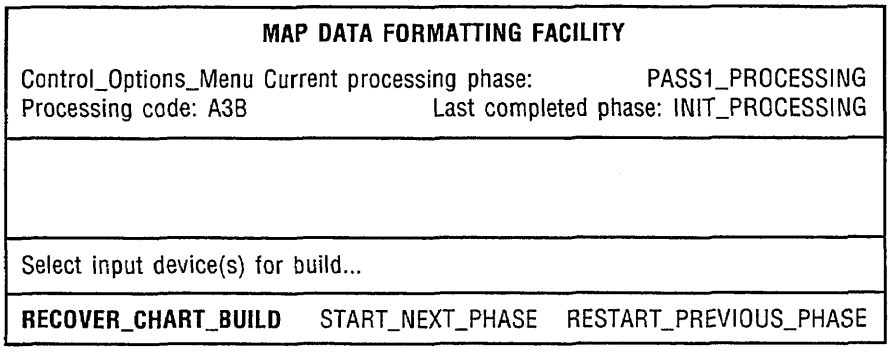



Fig. 44 — Control_Options menu recover build option selection

shifted to the segment and to compression in general. There may be some confusion because of the continued references to codebook(s) throughout the CAC processing system. Hence, it must be understood that the term *codebook* is often used synonymously with *compression* throughout the CAC processing system.

### *5.4.1 CHART_STATUS.DAT Values*

After DOWNSAMPLING has completed, BUILD_PASS1_PROCEDURES starts to build command procedures to compress the downsampled data. Chart_Control updates the CHART_STATUS status_record processing_step field as BUILD_ADRG_CB_PROCS and the last_completed_step field as PASS1_DOWNSAMPLE (Table 1.)

Sqmdff will list the processing queues that contain jobs named ###_BLDPASS1_**, where ### is the processing thread name and ** is the CDROM reader number.

These jobs execute quickly, so they will not appear in the MDFF_BATCH queue for long. Each job builds multiple compression command procedures in the MDFF_SCRATCH:[CD#*****] directory:

MDFF_SCRATCH: [CD#*****] CD#*****_xxxx.COM,

where

| | |
|---|---|
| # | is the one-digit Mastering Facility Code number, |
| ***** | is the five-character MDFF sequence number, and |
| xxxx | is the four-character sequential command procedure number. |

Each compression command procedure contains a list of TS segment files to be compressed and their respective compressed file names. During execution, the command procedure color-compresses the TS segment files by 3:1 (from 24 bits/pixel to 8 bits/pixel) and generates a compression codebook to compress the 8 bits/pixel data by 4:1. The procedure also generates a decompression codebook for each compressed segment file.

Chart_Control submits these command procedures to the generic codebook batch queue for execution (which builds compression codebooks) and updates the CHART_STATUS status_record processing_step field as SUBMIT_ADRG_CB_PROCS and the last_completed_step field as BUILD_ADRG_CB_PROCS (Table 1). When the codebook compression procedures are executed, Chart_Control updates the CHART_STATUS status_record processing_step field as BUILD_ADRG_CBS and the last_completed_step field as SUBMIT_ADRG_CB_PROCS (Table 1).

Each final CAC segment is one file; with the xxxxxxxx.xxz,

where

| | |
|---|---|
| xxxxxxxx.xx = | a 10-digit number based on the formula shown in Fig. 43 with a period "." placed between the eighth and ninth digits and |
| z = | a one-digit TS zone designation (0 = SP, 1= ST, 2 = EQ, 3 = NT, 4 = NP, and 5 = NP). |

Execute the DISPLAY_SEGMENTS_X utility to view downsampled segments that were compressed during PASS1 (Sec. 3.4.4).

### *5.4.2 Compression Errors: Detection and Handling*

### *5.4.2 Compression Errors: Detection and Handling*

This section describes how the utilities List_Chart_Status, Sqmdff, Sqcode, Diskuse, and List_Codebook_Status monitor compression.

Chart_Control writes compressed segments to the logical device CHART_ODI. The Diskuse utility monitors available disk space. The only way to regain disk space on CHART_ODI is to delete files off the disk volume set (e.g., delete data from *another processing thread* that has completed). Contact the system manager *before* disk space becomes a problem and impedes processing.

If List_Chart_Status shows that a CDROM entry is still in the compression phase, but Sqcode does not show any compression jobs in the codebook queues, then the system has suffered a power loss. If so, restart Chart_Control and select the RECOVER_CHART_BUILD option (Sec. 5.7.3.1).

List_Codebook_Status also monitors the codebook compression jobs as shown with the following command:

*list_codebook_status/cd* = xxxx (xxxx is the MDFF sequence number for the CD_ROM.

If the value for any entry under "stat" is not "99" (completed) and Sqcode does not show the entries in the codebook queues, then the system may have suffered a power loss. Restart Chart_Control and select the RECOVER_CHART_BUILD option (Sec. 5.7.3.1).

## 5.5 Legend Processing

Legend data for an ADRG CDROM are processed concurrently with the chart data. Once an ADRG CDROM is selected for processing, its status_header legend_status flag is set to LEGEND_START. After processing begins, this flag is set to LEGEND_PROCESSING.

The ADRG CDROM is only flagged as complete when both legend and chart data processing are finished. Since legend data files are small (compared to chart data), legend processing finishes first, but the legend_status flag is not set to LEGEND_COMPLETE until after the last_completed_step flag has been set to PASS1_COMPLETE in case there is a problem with chart data processing. In this way, the chart data processing can be restarted without reprocessing legend data. Finally, after all processing is complete, the last_completed_step flag is set to CD_COMPLETE and the CDROM is ready for review.

Legend image files are downsampled and color-compressed just like the chart data files, but legend data are not spatially compressed. Downsampled files are stored in the same directory as the DS_SEGMENTS.DAT file for the corresponding ADRG CDROM (Sec. 3.5.5). Each legend file has a color palette, and both the legend image and its color palette are stored in the CAC CD legend directory.

Two errors occasionally arise, occurring during legend processing: (1) an incomplete image on the ADRG CDROM and (2) inadequate disk space. The first error is beyond typical processing control (contact NRL for a solution). The second can be corrected only by making more disk space available.

Finally, due to space constraints, Chart_Control provides the option of processing partial legend data or all of the legend data for an ADRG CDROM. By default, there is legend data for *each chart*

in an ADRG image. Each JOG ADRG CDROM typically contains four charts; TLM-50 (Topographic Line Map) and TLM-100 typically contain six to nine charts. The amount of space required for legend data becomes *significant* (up to 40–60% of the resultant CAC). Therefore, we recommend processing legend data from only one representative chart per CDROM to ensure sufficient space for chart data. The "process_all_legend_data" flag, which controls this option, is automatically set during CHART_STATUS file creation: FALSE for TLM-50 and TLM-100 data and TRUE for all other scales.

### 5.6 Review Forms

Review forms are printed automatically as each ADRG CDROM completes processing. These forms are used during quality control checks. Use the following command to print additional review forms:

*Print/Que* = **print-queue**/*Noflag* **MDFF_SCRATCH:[CDx#####]REVIEW.LIS,**

where

print-queue = the name of a print queue,
x = a one-digit Mastering Facility Code, and
##### = a five-digit MDFF sequence number.

A review form must be completed for every ADRG CDROM in the CAC ODI build to facilitate the quality control of every compressed segment. Figure 45 shows the first four lines of a review form for processing thread A3B.

The review form also includes a section to quality-check legend data (Fig. 46).

### 5.7 Other Chart_Control Options

This section describes other options available from the PASS1 main menu during chart processing.

Reviewer: _____ Page 1 of ____

CD#: 538 # of codebooks: 20 Date created: 21-JUL-1995

Processing Code: A3B

NIMA_Stock_Number: ARC1 TPCXXK26D Date CHECKED:

| CHECKED | CB# | Stat | Pass | Map Zone | Pal Id | CB Used | MinRow | MinCol | MaxRow | MaxCol | Distort |
|---|---|---|---|---|---|---|---|---|---|---|---|
| _____ | 001 | 99 | 1 | 2 | 0130 | 1 | 35 | –346 | 39 | –342 | 177 |
| _____ | 002 | 99 | 1 | 2 | 0130 | 2 | 35 | –341 | 39 | –337 | 188 |
| _____ | 003 | 99 | 1 | 2 | 0130 | 3 | 35 | –336 | 39 | –332 | 114 |
| _____ | 004 | 99 | 1 | 2 | 0130 | 4 | 35 | –331 | 39 | –327 | 158 |

Fig. 45 — PASS1 review form

Reviewer: _____ Page 3 of ____

CD#: 538 # of codebooks: 20 Date created: 21-JUL-1995

Processing Code: A3B

NIMA_Stock_Number: ARC1 TPCXXK26D Date CHECKED:

Distribution rectangle: TPPM01

_____ 01 EL
_____ 01 GE
_____ 01 GL
_____ 01 GR
_____ 01 IN

Fig. 46 — Legend data portion of review form

**MAP DATA FORMATTING FACILITY**

PASS1_Menu Current processing phase: PASS1_PROCESSING
Processing code: A3B Last completed phase: INIT_PROCESSING

Select an option...

VALIDATE_READERS SELECT_READERS **BACKUP_ODI(Tue Jan 2 00:00:00 1996)**
MODIFY_GENERIC_CB_QUEUE CONTROL_OPTIONS

Fig. 47 — Selection of the BACKUP_ODI option

### *5.7.1 BACKUP_ODI Option*

The BACKUP_ODI option in Chart_Control (Fig. 47) backs up files that are associated with a given processing thread. Some of these include files within the CHART_SEGMENTS, CHART_ODI_DISK, and MDFF_SCRATCH directories. Files in the ADRGDIR## directories (where ## denotes CDROM reader) *are not backed up* since they are restored easily from their original ADRG CDROMs.

Chart_Control will prompt the user to verify the start of the backup procedure. Enter "Y" to continue or "N" (default). Chart_Control also prompts the user for the device name/number to use (see system manager). Enter the appropriate device number.

Chart_Control submits a command file to perform the backup in MDFF_BATCH queue. The backup command file is located in the MDFF_SCRATCH directory and has the following file naming convention:

BACKUP_###_FILES.COM (### = processing thread name).

When the backup completes, Chart_Control updates the *last_backup_time* field in the CHART_STATUS.DAT file and sends a mail message to users listed in the file MDFF_PROCESSORS.DIS.

Examine the backup log file to verify that the backup operation completed normally:

MDFF_SYSTEM:[BACKUP]BACKUP_###_FILES.LOG (### = processing thread name).

Remove the backup tape cartridge from the tape drive. Set the write-lock switch in the "lock" position and the label tape with the date of the backup and "BACKUP_###_FILES" (Fig. 20).

### *5.7.2 MODIFY_GENERIC_CODEBOOK_QUEUE Option*

The MODIFY_GENERIC_CODEBOOK_QUEUE option in Chart_Control modifies the codebook compression queue usage to balance the processing load on the system (Fig. 48).

Selecting this option results in a listing of all of the current codebook compression queues (Fig. 49). An editor is automatically started so that the user can modify, add, or remove queue names. Press ^Z to exit the editor after making any necessary changes. The menu status field will report that the CODEBOOK_QUEUES list has been modified.

These modifications are not permanent. The next time the system reboots, any changes would be lost. To make them permanent, Chart_Control automatically opens the file that defines these queues (at system startup) and permits the user to modify the queue names permanently (Fig. 50).

```
                    MAP DATA FORMATTING FACILITY
PASS1_Menu                          Current processing phase: PASS1_PROCESSING
Processing code: A3B                    Last completed phase: INIT_PROCESSING

Select an option...

VALIDATE_READERS   SELECT_READERS   BACKUP_ODI(Tue Jan 2 00:00:00 1996)
MODIFY_GENERIC_CB_QUEUE   CONTROL_OPTIONS
```

Fig. 48 — Selection of the MODIFY_GENRIC_CB_QUEUE option

```
*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*
*CAUTION*                                                  *CAUTION*
*CAUTION*     Any changes WILL be lost at reboot time      *CAUTION*
*CAUTION*                                                  *CAUTION*
*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*CAUTION*

Processing code: A3B
Current queues: CODEBOOK1,CODEBOOK2     <THE CURSOR IS AUTOMATICALLY PLACED HERE FOR THE
                                         USER TO ADD, DELETE, OR MODIFY CODEBOOK QUEUE NAMES.
```

Fig. 49 — Dynamic codebook compression queue modification

```
$       set noon
$!
$! Logical names to process A3B data.
$!
$       create/name_table/parent_table=mdff/attr=supersede -
            /prot=(s:rewd,o:rewd,g:rewd,w) A3B
$       set acl/object_type=logical_name_table A3B -
            /like=(object_name=mdffexe:acl_logical_example.txt,object_type=
$!
$       @mdffexe:use_adrg_logicals A3B
$!
$! Set processing code used in building command procedures
$       define/table=A3B processing_code A3B
$!
$! Set up generic codebook queue
$       define/table=A3B codebook_queues -
            CODEBOOK1,CODEBOOK2          < Modify this line of text to make your desired
                                         CODEBOOK queue specification permanent
$!
            $! Build scratch area
$       define/table=A3B mdff_scratch -
            /translation_attributes = (concealed) -
```

Fig. 50 — Edit the file DEFINE_xxx_LOGICALS.COM (xxx = processing thread name) to permanently change codebook queue names

| MAP DATA FORMATTING FACILITY | |
|---|---|
| Control_Options_Menu | Current processing phase: PASS1_PROCESSING |
| Processing code: A3B | Last completed phase: INIT_PROCESSING |
| | |
| Select function... | |
| RECOVER_CHART_BUILD START_NEXT_PHASE RESTART_PREVIOUS_PHASE | |

Fig. 51 — The Control_Options menu

### *5.7.3 The CONTROL_OPTIONS Menu*

The Control_Options menu in Chart_Control includes three options for maintaining the control of CAC processing through (Fig. 51):

- **RECOVER_CHART_BUILD** recovers processing after a power loss;
- **START_NEXT_PHASE** starts the next processing step; and
- **RESTART_PREVIOUS_PHASE** restarts a previous processing step.

#### 5.7.3.1 RECOVER_CHART_BUILD Option

Process_Chart is automatically resubmitted to MDFF_BATCH when Chart_Control is invoked unless the /NOSUBMIT qualifier to Chart_Control is used. Therefore, do not use this qualifier during recovery. Process_Chart *must be executing* for these recover procedures to work.

RECOVER_CHART_BUILD recovers processing after a power loss (Sec. 5.3.2.2) based on the value of the last_completed_step field for each CHART_STATUS.DAT status_record entry:

*NOOP*:

Processing of entries with this value will continue normally after Chart_Control is executed and Process_Chart is resubmitted to the MDFF_BATCH queue.

*COPY_ADRG*:

System failures typically cause entries to abort during this processing step. Examine the copy procedure log file in the MDFF_SCRATCH directory to determine if this is the case. Copy procedure log file names use the following format:

###_COPYADRG**.LOG (### = processing thread name, ** = CDROM reader number).

If there are no other failures noted in the log file, return to the RECOVER_CHART_BUILD procedure and enter "Y" when prompted "Restart from last completed step? [N]:." ADRG CDROM data will resume copying and processing will continue normally.

*PASS1_DOWNSAMPLE*:

Examine the downsample log file in the MDFF_SCRATCH area to determine how far processing had progressed. Downsample log file names using the following format:

###_DOWNSAMP**.LOG, where ### is the processing thread name and ** is the CDROM reader number.

If there was a system failure, enter "Y" when prompted "Restart from last completed step? [N]:." Next, the option to perform a *cold* or *warm restart* is given. A cold restart begins downsampling from scratch (i.e., all data for this ADRG CDROM are downsampled again). A warm restart resumes downsampling where the downsample job had terminated. Normally, a warm restart is performed unless a serious problem has occurred, in which case the problem must be resolved and then a cold restart can be performed.

*BUILD_ADRG_CB_PROCS*:

Enter "Y" when prompted "Restart from last completed step? [N]:."

*SUBMIT_ADRG_CB_PROCS*:

Use Sqcode to check the queues and determine whether any compression jobs were running for this ADRG CDROM. If there are jobs in the codebook compression queues, do not use the recover process, since recovery will resubmit all compression jobs again and cause multiple compression jobs to execute for the same codebook entry. Instead, allow the compression jobs to complete and call system support for additional instructions.

If no jobs are in the queues, enter "Y" when prompted "Restart from last completed step? [N]:."

*BUILD_ADRG_CBS*:

This is the only step where a restart from the previous step *should not be performed.* Enter "N" when prompted "Restart from last completed step? [N]:."

Answer "Y" to the next prompt:
"Restart codebooks for stock# (CDROM#) [N]:," where stock# is the NIMA ADRG stock number and CDROM# is the MDFF sequence number.

For example, "Restart codebooks for ARC5 NF3904 (CD001646) [N]:."

The recovery operation reads each entry in the CODEBOOK_STATUS.DAT file and performs a certain action based on the entry's status:

- When the entry status equals 0 or 77, the compression command file is resubmitted for processing.
- When the entry status equals 1, the compression log file is read. The status is then changed to either 77 (if the job aborted) or 99 (if it completed successfully). If the entry status is changed to 77, the preceding step will have to be executed again to resubmit the compression command file.

These steps avoid redundant compressions, since some downsampled data were already compressed before the system failed. Unless an error occurred during downsampling, there is no reason to compress the data again. Review the downsample log file to determine whether an error occurred. If so, the ADRG information (i.e., RGB image data) could have been inadvertently deleted from the ADRGDIRxx:[FILES] directory (xx = CDROM reader number). When ADRG information is deleted, the CDROM must be revalidated and recopied from scratch.

### 5.7.3.2 START_NEXT_PHASE Option

START_NEXT_PHASE advances processing to the next step (and usually to another phase). As shown in Fig. 52, Chart_Control displays the next processing step and a prompts the user to proceed by highlighting one of the available choices. To avoid an *unintentional* indication to proceed, Chart_Control verifies a "Y" reply before advancing to the next processing step.

A message is displayed as a reminder that processing will be advancing to an intermediate step in PASS2. A Wait_State menu is provided while processing is in an intermediate step, regardless of the processing phase (Fig. 53).

The Wait_State menu provides the following three options that are listed and described below:

- **EXIT** terminates program execution.
- **UPDATE_SCREEN** checks the current CHART_STATUS values. If an intermediate step is no longer listed, the appropriate menu will replace the Wait_State menu.
- **BACKUP_ODI** performs a backup operation.

| **MAP DATA FORMATTING FACILITY** | |
|---|---|
| Start_Next_Phase | Current processing phase: PA hSS1_PROCESSING |
| Processing code: A3B | Last completed phase: INIT_PROCESSING |
| **The next processing step will become PASS2_BUILD_SEGMENTS_FILE**<br>**The last completed step will become PASS1_PROCESSING**<br>NOTE: Control will be advancing to intermediate processing steps. | |
| Do you want to start the next processing phase? | |
| NO YES | |

Fig. 52 — Start_Next_Phase menu

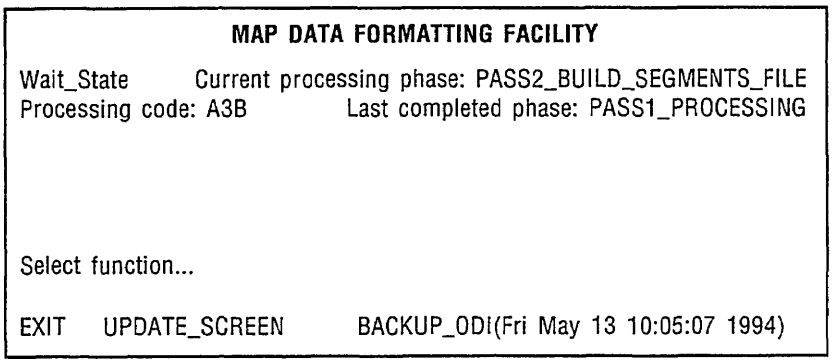



Fig. 53 — Wait_State menu

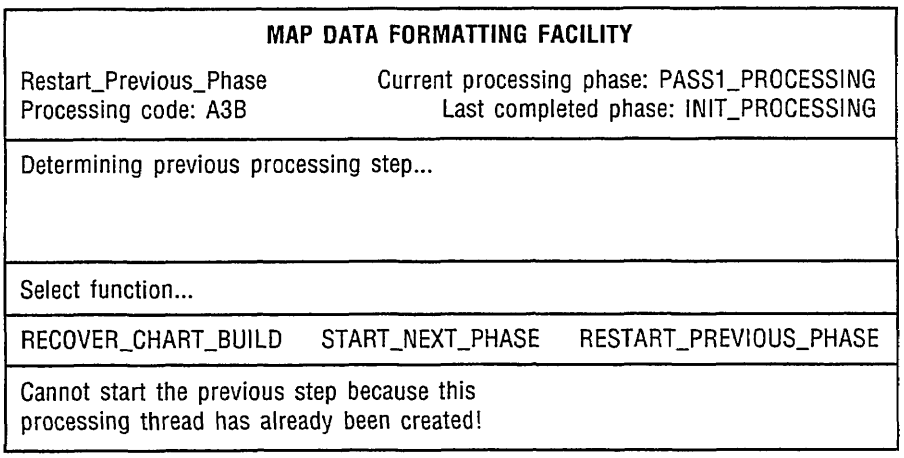



Fig. 54 — RESTART_PREVIOUS_PHASE menu

#### 5.7.3.3 RESTART_PREVIOUS_PHASE Option

RESTART_PREVIOUS_PHASE returns processing to a previous step even if it is in another phase (Fig. 54). Processing cannot return to the INITIALIZATION phase since this processing thread already exists, so this option is limited for PASS1.

## 6.0 PASS2 PROCESSING

### 6.1 Introduction

The PASS2 processing phase defines the CAC ODI boundaries and fills edge segments with downsampled data from surrounding ADRG CDROMs. Chart_Control executes PASS2 through two menus: an ODI Trim menu that defines the CAC ODI bounds for trimming excess data, and a PASS2 menu that provides options to include surrounding ADRG CDROMs and fill edge segments, perform backup operations, and control processing. Only downsampling occurs in PASS2; compression is performed in PASS3.

This section describes segment classification, how segments are filled with data, and individual PASS2 processing steps. Table 2 provides the status_record values (in CHART_STATUS.DAT) for each PASS2 processing step.

Table 2 — Values of Status Record Control Fields ("Processing_Step" and "Last_Completed_Step") in CHART_STATUS.DAT File During PASS2 Processing. Values are Provided for Each ADRG CDROM Being Processed for a Given Thread.

| DESCRIPTION OF PASS2 PROCESSING STEPS | PROCESSING_STEP | LAST_COMPLETED_STEP |
|---|---|---|
| **Preparing Final Image Boundaries and Trimming** | | |
| Building SEGMENTS.DAT file (Sec. 6.3) | PASS2_BUILD_SEGMENTS_FILE | PASS1_PROCESSING |
| Building ODI_BOUNDS.DAT file (Sec. 6.4) | PASS2_BUILD_ODI_BOUNDS_FILE | PASS2_BUILD_SEGMENTS_FILE |
| Defining trim boundaries for final image file (Sec. 6.5) | PASS2_DEFINE_TRIM_BOUNDS | PASS2_BUILD_ODI_BOUNDS_FILE |
| Trimming segments from the image file (Sec. 6.6) | PASS2_TRIM_SEGMENTS | PASS2_DEFINE_TRIM_BOUNDS |
| **Filling and Downsampling Edge Segments** | | |
| Identifying ADRG data to fill edge segments (Sec. 6.7) | PASS2_FIND_EDGEFILL_CDROMS | PASS2_TRIM_SEGMENTS |
| Filling edge segments with all available data (Sec. 6.8) | PASS2_PROCESSING | PASS2_FIND_EDGEFILL_CDROMS |
| Downsampling edge segment data (Sec. 6.8.1) | PASS2_DOWNSAMPLE | PASS2_PROCESSING |
| Immediately after PASS2 downsampling has completed (Sec. 6.8.1) | NO_OP | PASS2_DOWNSAMPLE |
| **Decompressing segments for a CAC Update (only during UPDATE)** | | |
| "Faux" downsampling segments for a CAC ODI update (Sec. 6.9) | PASS2_FAUX_DOWN-SAMPLE | PASS2_PROCESSING |

## 6.2 Segment Classification: Core and Edge Segments

After the PASS1 phase has completed, all segments are classified as either core or edge as described in Sec. 2.4 and illustrated in Fig. 2.

Core segments were compressed in PASS1. In contrast, edge segments are not usually filled in PASS1, since they lie on a chart's edge and usually need data from an adjacent chart(s) to be completely filled. The only edge segments that can he filled with data in PASS1 are edge segments that do not extend beyond the chart boundary by even one pixel. All other edge segments are filled in PASS2.

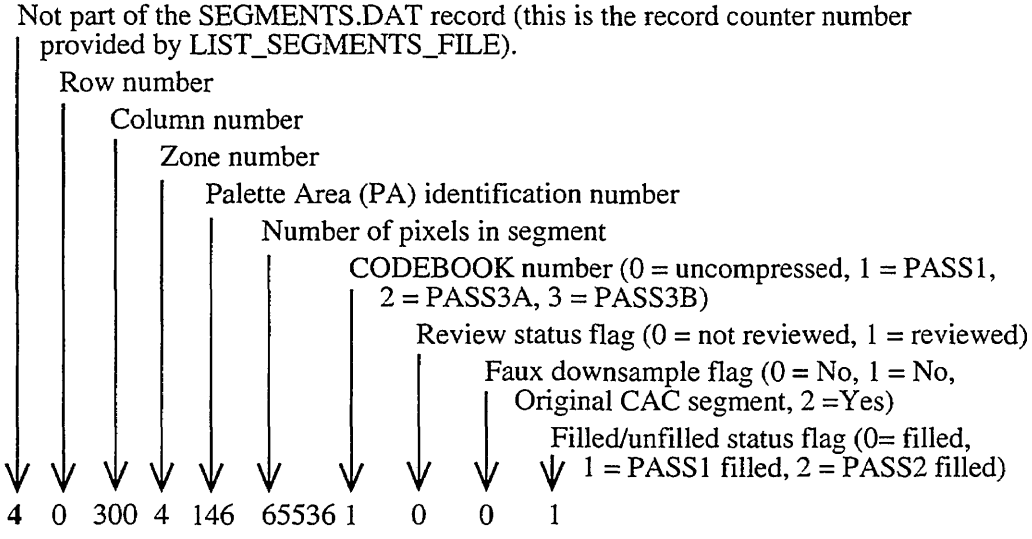
As shown in Fig. 2, edge segments are split into two categories during PASS2: *unfilled* and *filled* segments. Filled segments are completely filled with data during PASS2 because the necessary adjacent ADRG chart was available. Unfilled edge segments are only partially filled due to the unavailability of surrounding ADRG. Figure 4 shows the difference between processing core and

edge segments. Run DISPLAY_SEGMENTS_X to display core and edge segments on an X terminal (Sec. 3.4.4).

## 6.3 PASS2_BUILD_SEGMENTS_FILE

The first PASS2 processing step is *PASS2_BUILD_SEGMENTS_FILE* (Table 2). Process_Chart creates the file SEGMENTS.DAT in the MDFF_SCRATCH:[000000] to differentiate edge and core segments. SEGMENTS.DAT maintains a list of each segment's row and column values and categorizes each segment as core, filled edge, or unfilled edge.

Figure 55 lists the SEGMENTS.DAT file. This listing was generated with the LIST_SEGMENTS_FILE utility:




| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 26,282 segments | | | | | | *The first record indicates the total number of segments.* | | | | |
| | | | | | | *The remaining records are sequentially numbered.* | | | | |
| 1 | 0 | 297 | 4 | 146 | 53538 | 3 | 0 | 0 | 0 | *Each segment is represented by a record, this is the first record.* |
| 2 | 0 | 298 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 3 | 0 | 299 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 4 | 0 | 300 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 5 | 0 | 301 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 6 | 0 | 302 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 7 | 0 | 303 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| . | | | | | | | | | | |
| . | | | | | | | | | | |
| 26277 | –1 | 370 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 26278 | –1 | 371 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 26279 | –1 | 372 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 26280 | –1 | 373 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 26281 | –1 | 374 | 4 | 146 | 65536 | 1 | 0 | 0 | 1 | |
| 26282 | –1 | 375 | 4 | 146 | 65536 | 2 | 0 | 0 | 2 | *This is the 26,282nd record* |

Fig. 55 — Sample listing of a SEGMENTS.DAT file

## 6.4 PASS2_BUILD_ODI_BOUNDS_FILE

In this step, PROCESS_CHART creates the file ODI_BOUNDS.DAT (stored in MDFF_SCRATCH) that contains CAC ODI boundaries for each TS zone. ODI_BOUNDS.DAT is used to trim extraneous areas of coverage.

The segments processed in PASS1 define the initial area of CAC ODI coverage. PASS2 provides a CHART_ODI_TRIM menu to modify ODI_BOUNDS.DAT if the ODI boundaries need to be trimmed.

The List_ODI_Bounds utility lists ODI_BOUNDS.DAT for the currently defined processing thread. Two latitude/longitude coordinates define the northeast and southwest corners of a rectangular boundary area (Fig. 56). There is one pair of coordinates for every contiguous area of data in each TS zone of the CAC ODI (Fig. 57).

The following is an example of an ODI_BOUNDS.DAT file (listed with the List_ODI_Bounds utility) with boundaries for an area of coverage in the North Polar Zone:

```
$ LIST_ODI_BOUNDS

Zone: 4
minlon: 2.759003
maxlon:18.729025
minlat: 59.878157
maxlat: 68.137002
```

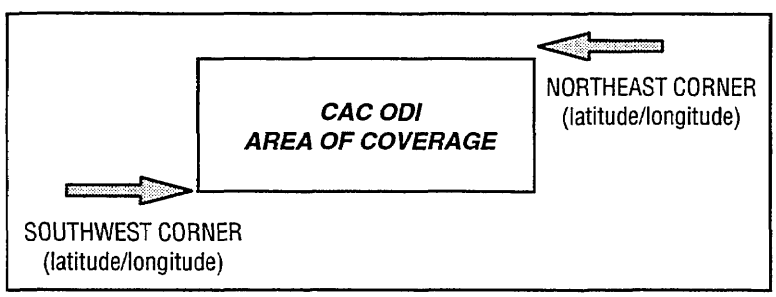



Fig. 56 — Rectangular boundary area for a CAC ODI bounds set

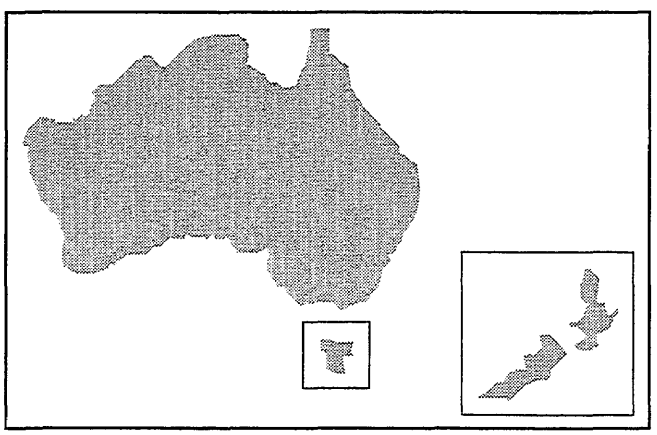


Fig. 57 — PASS2 CAC ODI areas of coverage

Chart_Control automatically advances to the next processing step after building the ODI_BOUNDS.DAT file.

## 6.5 PASS2_DEFINE_TRIM_BOUNDS

This step defines the final CAC ODI area of coverage and modifies the ODI_BOUNDS.DAT file to contain all desired rectangular boundaries for each TS zone in the CAC ODI.

The ODI_BOUNDS.DAT file is modified through the CHART_ODI_TRIM menu (Fig. 58). Each TS zone with one or more bound sets of data is displayed so that each of the zone's bound sets may be accepted, modified, or deleted. For example, Fig. 58 presents a set of bounds for the NP zone. The menu provides options for accepting, modifying, or deleting these coordinates. Once all existing bound sets have been handled, the PASS2 Trim Menu prompts the user for any additional bound sets for this zone.

The PASS2_DEFINE_TRIM_BOUNDS process repeats these steps for each TS zone in the CAC ODI. After all zone bounds have been confirmed, control advances to the next CHART_ODI_TRIM menu (Fig. 59), which provides options to perform an ODI backup or switch to the CONTROL_OPTIONS menu.

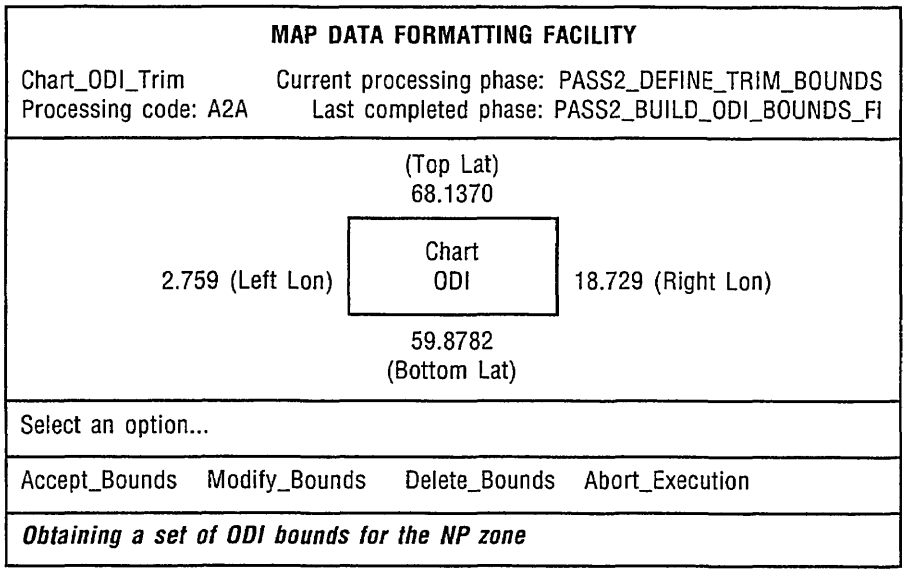



Fig. 58 — CHART_ODI_TRIM menu

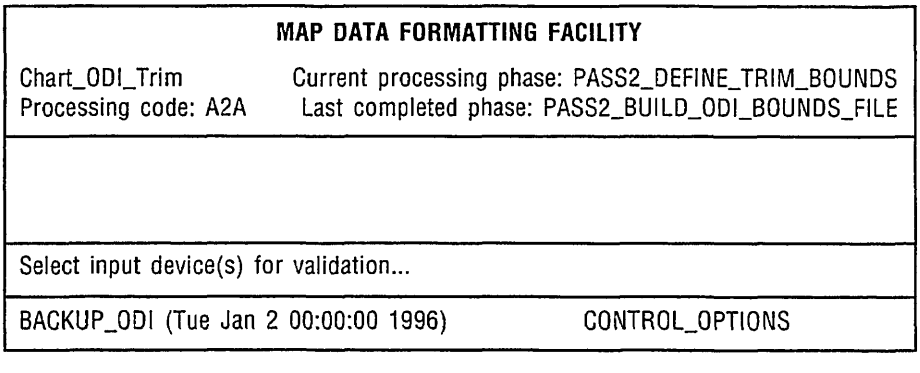



Fig. 59 — CHART_ODI_TRIM menu after ODI bounds selection and confirmation

An ODI backup is recommended at this time, since trimming may have removed files from this processing thread. The CONTROL_OPTIONS menu advances to the next processing step (PASS2_TRIM_SEGMENTS), which performs the ODI trim. In subsequent Chart_Control jobs, the CHART_ODI_TRIM menu will continue to be provided *until control is advanced* to the next processing step with the CONTROL_OPTIONS menu.

## 6.6 PASS2_TRIM_SEGMENTS

This processing step uses the boundaries that were selected and verified in the preceding step to trim the CAC ODI.

A WAIT_STATE menu is provided, since this an intermediate processing step. Control will automatically advance to the next processing step (PASS2_FIND_EDGEFILL_CDROMS) once the ODI trim is complete.

## 6.7 PASS2_FIND_EDGEFILL_CDROMS

This processing step attempts to identify all unfilled edge segments. The process creates a list of all available ADRG CDROMs that are adjacent to each ADRG CDROM in the CHART_STATUS file. This list is named PASS2_CDROM_LIST and it is used in the next processing step (PASS2_PROCESSING) to fill the unfilled edge segments. The WAIT_STATE menu is provided again, since this is another intermediate processing step. Control will automatically advance to PASS2_PROCESSING once the PASS2_CDROM_LIST file is built.

## 6.8 PASS2_PROCESSING

This section describes the subtasks associated with the PASS2_PROCESSING step. This step is used primarily for filling edge segments. Section 6.8.1 describes the PASS2 processing tasks and menus and Sec. 6.8.2 describes different types of chart coverage areas and different methods of filling edge segments.

### *6.8.1 Processing Tasks and Menus*

The main PASS2 processing menu (Fig. 60) provides options to include additional ADRG CDROMs into the processing thread, perform an ODI backup, and provide services to maintain and control other aspects of chart processing (e.g., queue management).

Use the PRINT_CDROM_LIST option to print the current listing of available ADRG CDROMs for filling edge segments, identify required ADRG CDROMs, and then use the MODIFY_CDROM_LIST option (Fig. 61) to designate which entries are to use.

Scroll though the list of available ADRG CDROMs or use the GOTO_CDROM option to go directly to a list entry via the CDROM's NIMA stock number. Every ADRG CDROM will be flagged with one of the following status values shown in Fig. 62.

Initially, all entries have a "?" status. Before processing advances, entries should have either a "U" or "X" status (although it is not required) since this indicates that all possible CDROMs have been considered.

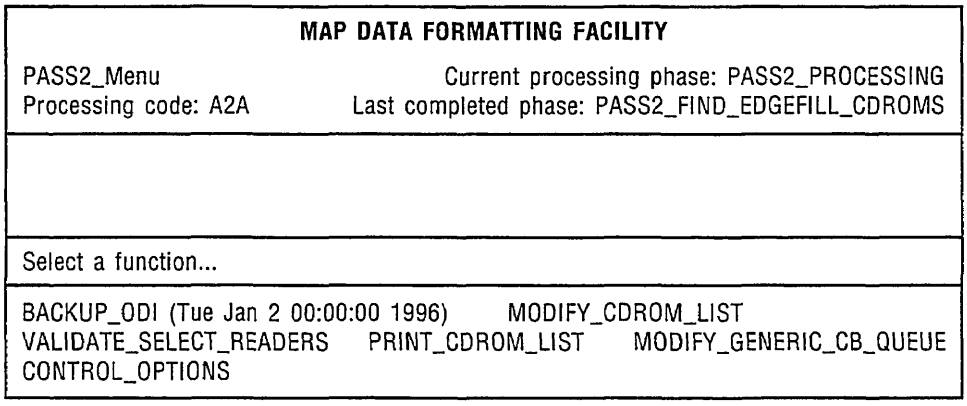



Fig. 60 — PASS2 main menu

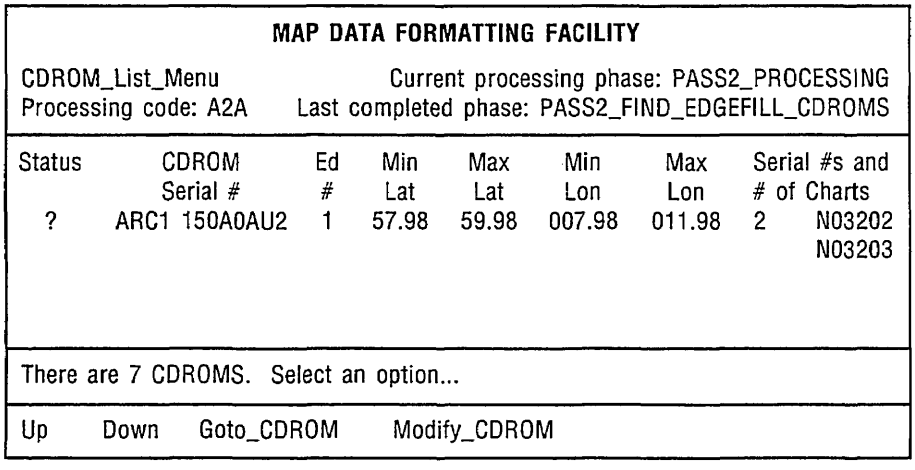



Fig. 61 — MODIFY_CDROM_LIST menu

| STATUS | MEANING |
|---|---|
| ? | Unknown |
| U | Allow this CDROM for use |
| X | Do not use this CDROM |

Fig. 62 — ADRG CDROM status value meanings

Once the list is complete, start filling edge segments by loading the selected ADRG CDROMs into CDROM readers and using the VALIDATE_SELECT_READERS option (Fig. 63). **Highlight** each CDROM reader with an ADRG CDROM to be used and repeat this validation/selection for every ADRG CDROM to be used for edge filling.

All selected readers will have corresponding PASS2 downsampling jobs submitted to the MDFF_BATCH queue. These ADRG CDROMs will be added to the CHART_STATUS.DAT file with a status_record last_completed_step value PASS2_DOWNSAMPLE (Table 2).

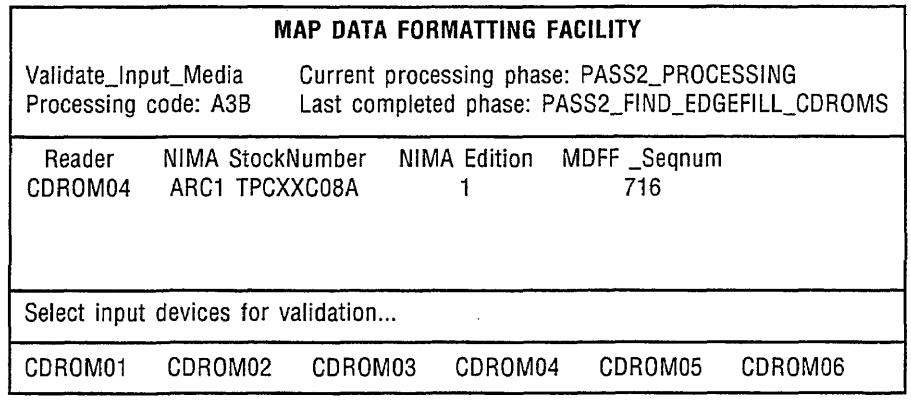



Fig. 63 — VALIDATE_SELECT_READERS menu

Check the status of PASS2 entries with the command:

*list_chart_status/pass = 2.*

There is no error recovery for PASS2 downsampling. If an error occurs, use the VALIDATE_SELECT_READERS menu to reselect the ADRG CDROM and repeat downsampling.

After all necessary CDROMs have been downsampled, use the CONTROL_OPTIONS menu to advance processing to the next processing step (PASS2_FAUX_DOWNSAMPLE).

### *6.8.2 Segment Filling*

This section describes different types of chart areas coverage and different methods of filling edge segments. The typical chart area of coverage is rectangular (Fig. 64a), but many coverages are not. Some charts may be incomplete; i.e., they are missing some data inside the designated area of coverage (Fig. 64b). Figure 64a–e provides examples of how chart coverage areas can vary. In rare cases, the use of incomplete charts (Fig. 64b) results in having unfilled inner edge segments.

## 6.9 PASS2_FAUX_DOWNSAMPLE

This is the final step in PASS2 processing, but it is only used when a CAC ODI UPDATE is performed. During this step, CAC segments with a value of 2 in the repair flag field of the SEGMENTS.DAT file, which were copied from CDROM during INITIALIZATION, are decompressed. The WAIT_STATE menu is displayed since this an intermediate processing step. After all these segments are decompressed, control automatically advances to the PASS3 Phase.

# 7.0 PASS3 PROCESSING

## 7.1 Introduction

This phase compresses all of the remaining uncompressed filled segments (phase PASS3A) and then unfilled segments (phase PASS3B). At the completion of PASS3, all segments that are included

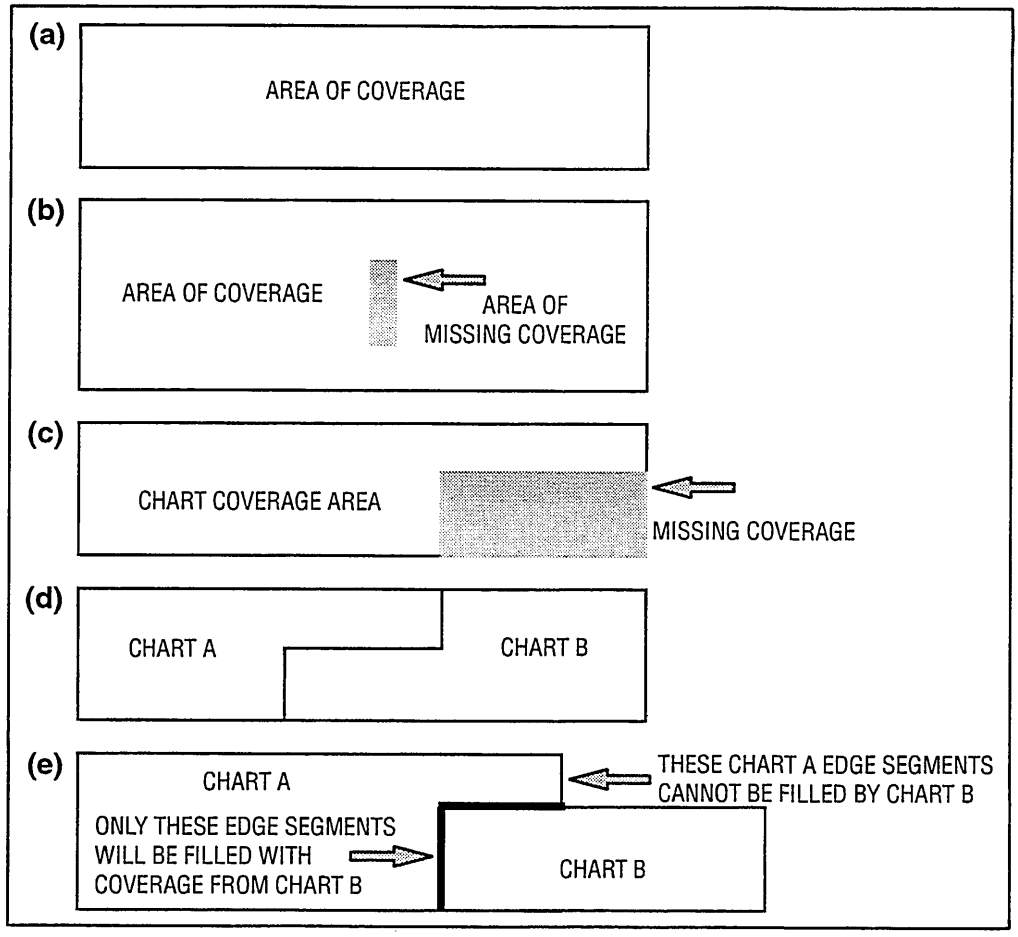



Fig. 64—(a) Full rectangular actual area of coverage; (b) incomplete chart (note area of missing data within the chart that results in unfilled inner edge segments); (c) nonrectangular actual area of coverage; (d) two charts with nonrectangular areas of coverage that can be fit together to form a full rectangular area of coverage; and (e) charts with nonrectangular areas of coverage that can be filled by some of the edge segments from other charts.

in the CAC ODI will have been compressed. This section describes the PASS3 menu and all associated steps. Table 3 provides the status_record values from CHART_STATUS for each PASS3 step.

## 7.2 PASS3_VERIFY_DS_SEGMENTS

This is the first step in the PASS3 phase, which verifies the downsampled segments file SEGMENTS.DAT for accuracy. After verifying that the SEGMENTS.DAT file is error free, control automatically advances to the next processing step. If errors are detected during verification, the CHART_STATUS flag *verify_ds_segs_flag* is set to false so the errors may be identified and corrected. After errors are corrected, Chart_Control displays the VERIFY_DS_SEGS menu (Fig. 65) and repeats verification. Control advances to PASS3_BUILD_FILLED_PROCS only after verification occurs without any errors.

Table 3 — Values of Status Record Control Fields ("Processing_Step" and "Last_Completed_Step") in CHART_STATUS.DAT File During PASS3 Processing. Values are Provided for Each ADRG CDROM Being Processed for a Given Thread.

| DESCRIPTION OF PASS3 PROCESSING STEPS | PROCESSING_STEP | LAST_COMPLETED_STEP |
|---|---|---|
| **Compressing Filled Edge Segments** | | |
| Verifying SEGMENTS.DAT for accuracy (Sec. 7.2) | PASS3_VERIFY_DS_ SEGMENTS | PASS2_FAUX_ DOWNSAMPLE |
| Building procedures to compress filled edge segments (Sec. 7.3) | PASS3_BUILD_FILLED_ PROCS | PASS3_VERIFY_DS_ SEGMENTS |
| Submitting procedures to compress filled edges (Sec. 7.4) | PASS3_SUBMIT_FILLED_ PROCS | PASS3_BUILD_FILLED_ PROCS |
| Compressing filled edge segments (Sec. 7.5) | PASS3_COMPRESS_FILLED | PASS3_SUBMIT_FILLED_ PROCS |
| **Compressing Unfilled Edge Segments** | | |
| Building procedure to compress unfilled edge segments (Sec. 7.6) | PASS3_BUILD_UNFILLED_ PROCS | PASS3_COMPRESS_FILLED |
| Submitting and monitoring compression of unfilled edges (Sec. 7.7) | PASS3_COMPRESS_ UNFILLED | PASS3_BUILD_UNFILLED_ PROCS |
| Providing control options and backup during PASS3 (Sec. 7.8) | PASS3_PROCESSING | PASS3_COMPRESS_ UNFILLED |

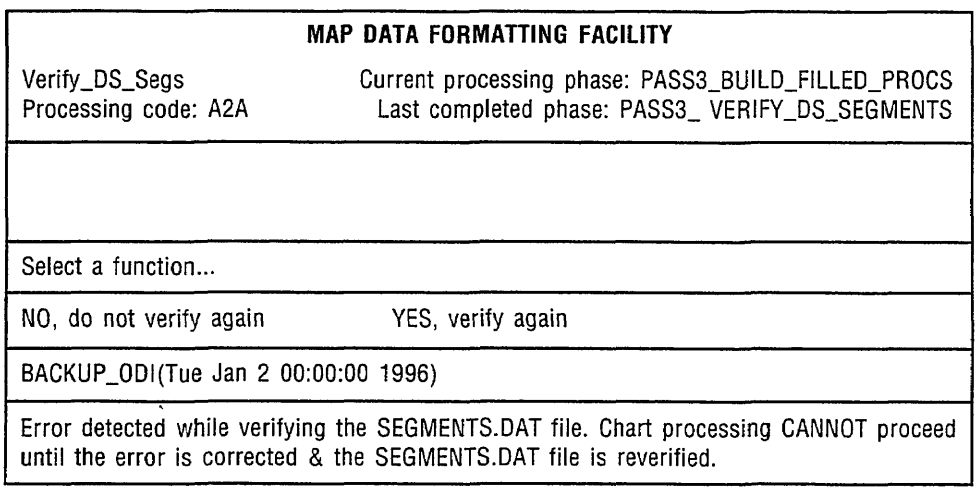



Fig. 65 — VERIFY_DS_SEGS menu

### 7.3 PASS3_BUILD_FILLED_PROCS

This step builds command procedures to compress all of the filled edge segments that are not yet compressed. Each procedure uses the palette (PA) number for the scale and zone in which they reside. The following procedures are built in the MDFF_SCRATCH:[PA****] directory and utilize the following file naming convention: MDFF_SCRATCH:[PA****]CD****_xxxx.COM,

where **** is the four-character PA number and
#### is the four-character sequential command procedure number.

Monitor the status of these procedures with the List_Codebook_Status utility:

*list_codebook_status/pa* = ****, where **** is the PA ID.

Since this is an intermediate step, the WAIT_STATE menu is provided. After all procedures are built, control automatically advances to the next processing step.

### 7.4 PASS3_SUBMIT_FILLED_PROCS

This step submits the command procedures that were built in the previous step. The WAIT_STATE menu is again provided for the duration of this intermediate step. As soon as these procedures are complete, control automatically advances to the next processing step.

### 7.5 PASS3_COMPRESS_FILLED

This step monitors the submitted command procedures (in the codebook queues) while they compress all uncompressed filled segments. Since it is an intermediate step, the WAIT_STATE menu is provided. As soon as these compression procedures are complete, control automatically advances to the next processing step.

Use the List_Codebook_Status utility to examine the status of the batch jobs in the compression queue:

*list_codebook_status/pa* = xxxx, where xxxx = color PA number.

### 7.6 PASS3_BUILD_UNFILLED_PROCS

This step builds a command procedure to compress all the unfilled segments. Since it is an intermediate step, the WAIT_STATE menu is provided. As soon as this command procedure is built, control automatically advances to the next processing step.

### 7.7 PASS3_COMPRESS_UNFILLED

This step builds, executes, and monitors the command procedure MDFF_SCRATCH:[000000]COMPRESS_FINAL_SEGMENTS.COM, which compresses all of the unfilled segments. Since it is an intermediate step, the WAIT_STATE menu is provided. As soon as this procedure is complete, control automatically advances to the next processing step.

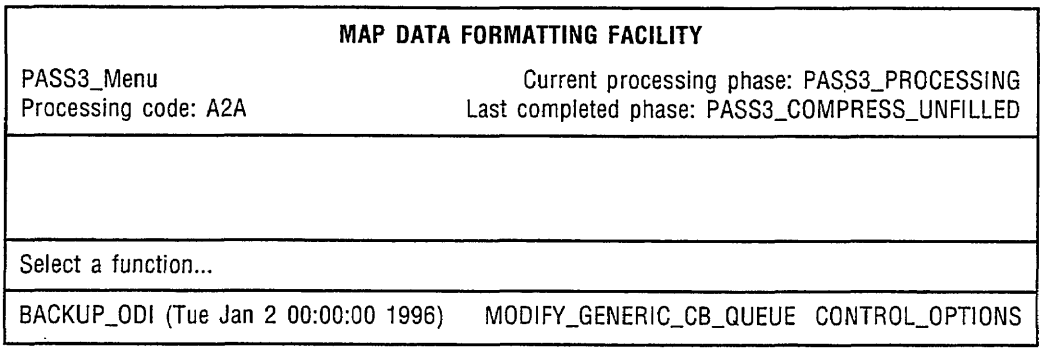



Fig. 66 — PASS3 processing menu

## 7.8 PASS3_PROCESSING

This is the final PASS3 step. The PASS3 processing menu (Fig. 66) provides options to control processing (i.e., CONTROL_OPTIONS) and to perform a CAC ODI backup. The CONTROL_OPTIONS menu advances control to the next processing phase (WRAPUP). However, the operator is responsible for reviewing all compressed segments with the CAC_QC utility before advancing to WRAPUP.

## 7.9 PASS3 Error Recovery

### *7.9.1 Recovery in PASS3A*

Process_Chart performs PASS3A error recovery when the value of the current processing step (in CHART_STATUS) is PASS3B_COMPRESS_FILLED and at least one compression job failed due to system failure.

To perform PASS3A recovery, first use the List_Codebook_Status utility to review the CODEBOOK_STATUS.DAT file (Sec. 3.5.4) and identify which codebook compression jobs have either not completed or have failed, and note the following:

(A) If a codebook entry has a status of "0" (meaning no action has been taken), check the compression queues with the Sqcode utility to determine if any other compression jobs are running for this codebook:

- If the queue(s) has stopped and jobs have been submitted, restart the queue(s) with the command (see Sec. 3.6 for additional information on queue management):

  *restart/queue* **queue name.**

- If a job for this entry is not listed in the queue, it may have failed (during execution) before its status could be updated. Use the Chart_Control/kill option to resume processing, as described later in this section.

(B) If the codebook entry status is "77" (meaning this job has failed), examine its associated log file in the MDFF_SCRATCH:[PAXXXX] directory:

PApppp_eeee.LOG, where pppp is a four-digit palette identification number and eeee is a four-digit entry number in the corresponding palette CODEBOOK_STATUS.DAT file.

For example, PA0125_0003.LOG is the log file for PA0125, CODEBOOK_STATUS.DAT entry 0003. Perform a recovery operation after reviewing the log file and the correcting the problem.

Figure 67 shows a CODEBOOK_STATUS.DAT file listing with examples of these errors. This listing contains 28 codebook entries: codebook entry 3 contains a "77" status, entry 28 contains a "0" status.

Use the following Chart_Control/kill command option to access the recovery menu in PASS3A:

*Chart_Control/kill.*

This option enables CAC processors and operators to determine the proper course of action and resume processing. Once the KILL option is invoked, Process_Chart is deleted from the MDFF_BATCH queue, the CHART_STATUS status_header regression flag is set to "KILL_PROCESSING," and a specialized version of the CONTROL_OPTIONS menu is provided (Fig. 68).

The RECOVER_CHART_BUILD option (Fig. 69) recovers processing from problems that halted PASS3A. When this option is selected in PASS3, Chart_Control examines each entry in the CODEBOOK_STATUS.DAT file and performs the following:

- If the entry status equals "0" or "77," Chart_Control submits the compression command file is submitted to batch.

*** List of CODEBOOK_STATUS file for PA# 125, 28 entries ***

| CB# | Stat | Map Pass | Pal Zone | Id | CB_To Use | MinRow | MinCol | MaxRow | MaxCol | Distort | Last_Seg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 001 | 99 | 1 | 0 | 0000 | 0 | 00000 | 00000 | 00000 | 00000 | 338 | 0166410522 |
| 002 | 99 | 1 | 0 | 0000 | 0 | 00000 | 00000 | 00000 | 00000 | 353 | 0166572531 |
| 003 | 77 | 1 | 0 | 0000 | 0 | 00000 | 00000 | 00000 | 00000 | 0 | 0000000000 |
| . | | | | | | | | | | | |
| . | | | | | | | | | | | |
| . | | | | | | | | | | | |
| 027 | 99 | 1 | 0 | 0000 | 0 | 00000 | 00000 | 00000 | 00000 | 290 | 0169974720 |
| 028 | 0 | 0 | 0 | 0000 | 0 | 00000 | 00000 | 00000 | 00000 | 0 | 0000000000 |

Fig. 67 — Example of CODEBOOK_STATUS.DAT file with codebook errors

**MAP DATA FORMATTING FACILITY**

Control_Options_Menu Current processing phase: PASS3_COMPRESS_FILLED
Processing code: A2A Last completed phase: PASS3_SUBMIT_FILLED_PROCS

Select function...

RECOVER_CHART_BUILD RESTART_PREVIOUS_PHASE

NOTE: This is a KILL RECOVERY.
Select the step where processing will resume.

Fig. 68 — Example of Control_Options menu for PASS3A recovery

| **MAP DATA FORMATTING FACILITY** | |
|---|---|
| Control_Options_Menu<br>Processing code: A2A | Current processing phase: PASS3_COMPRESS_FILLED<br>Last completed phase: PASS3_SUBMIT_FILLED_PROCS |
| | |
| Recover build? [N]: **Y** | |
| **RECOVER_CHART_BUILD** RESTART_PREVIOUS_PHASE<br><br>NOTE: This is a KILL RECOVERY.<br>Select the step where processing will resume. | |

Fig. 69 — The Control_Options menu with the RECOVER_CHART_BUILD option selected

| **MAP DATA FORMATTING FACILITY** | |
|---|---|
| Control_Options_Menu<br>Processing code: A2A | Current processing phase: PASS3_COMPRESS_FILLED<br>Last completed phase: PASS3_SUBMIT_FILLED_PROCS |
| | |
| Restart codebooks for (PA0125) [N]: **Y** | |
| RECOVER_CHART_BUILD RESTART_PREVIOUS_PHASE | |
| NOTE: This is a KILL RECOVERY.<br>Select the step where processing will resume. | |

Fig. 70 — Option to restart codebooks for PA0125

- If the status of the entry is "1," Chart_Control examines the compression log file and changes the status to "77" if the job aborted or to "99" if it completed successfully. If the entry status is changed to "77," execute the RECOVER_CHART_BUILD again to submit the compression command file to the compression queue (as shown in Fig. 70).

Next, Chart_Control will prompt the user for the job's priority for restart (100, 150, 200). Enter one of the optional priorities (priority 100 is commonly used); priority 200 will give this job top priority.

Now, check the codebook queues with the Sqcode utility to verify that all failed jobs have been resubmitted. Figure 71 provides a sample Sqcode listing that includes the compression jobs that were resubmitted after recovery.

Next, select the RESTART_PREVIOUS_PHASE option (Fig. 72) to return to a previous processing step (even if it is in another phase).

Generic batch queue A2A_CODEBOOKS

Batch queue CODEBOOK1, available, on VOODOO::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 55 | A2AP01250003 | CAC | Executing |
| 57 | A2AP01250028 | CAC | Executing |

Fig. 71 — Compression jobs resubmitted after recovery (Sqcode listing)

**MAP DATA FORMATTING FACILITY**

Control_Options_Menu Current processing phase: PASS3_COMPRESS_FILLED
Processing code: A2A Last completed phase: PASS3_SUBMIT_FILLED_PROCS

Select function...

RECOVER_CHART_BUILD **RESTART_PREVIOUS_PHASE**

NOTE: This is a KILL RECOVERY.
Select the step where processing will resume.

Fig. 72 — Control_Options menu for PASS3A: RESTART_PREVIOUS_PHASE

### *7.9.2 Recovery in PASS3B*

Process_Chart performs the PASS3B error recovery when the value of the current_processing_step (in CHART_STATUS) is PASS3_COMPRESS_UNFILLED. Process_Chart spawns a subprocess for this task, and the user cannot perform any other processing operation until this subprocess is finished.

The RECOVER_CHART_BUILD option is not valid for PASS3B. If the system loses power, Chart_Control automatically resubmits Process_Chart to the MDFF_BATCH queue and resumes processing. The Chart_Control/kill option is used in PASS3B when the user wants to restart processing at a previous step. As in PASS3A error recovery, use the Chart_Control/kill option to gain control and restart processing at a previous step.

## 8.0 WRAPUP PROCESSING

At the completion of PASS3, all of the segments have been compressed. The next step is WRAPUP, which purges and renames files, builds ancillary files, and verifies that all required files exist.

The WAIT_STATE menu is provided during WRAPUP since intermediate processing steps are used. WRAPUP provides a user menu after completing these intermediate processing steps. For

continuity, this section begins with the final PASS3 menu and describes considerations for WRAPUP before describing the actual steps that occur as WRAPUP is performed.

## 8.1 Preliminary Considerations

The user must perform a final review of the CAC ODI to verify that the CAC ODI is acceptable before advancing to WRAPUP. Use CAC_QC (App. B) to perform this quality control review. In addition, a backup is recommended since files will be modified (see Sec. 3.8.1 for additional information on backups).

After performing the final review and backup, advance processing to the WRAPUP phase. Select the CONTROL_OPTIONS, START_NEXT_PHASE option (Sec. 5.6.3) from the PASS3 menu (Fig. 73) to advance to WRAPUP.

Once control advances to WRAPUP it *cannot be restarted* at an earlier step (i.e., the CONTROL_OPTIONS RESTART_PREVIOUS_PHASE option is no longer available). As a double check, a verification prompt is issued (Fig. 74).

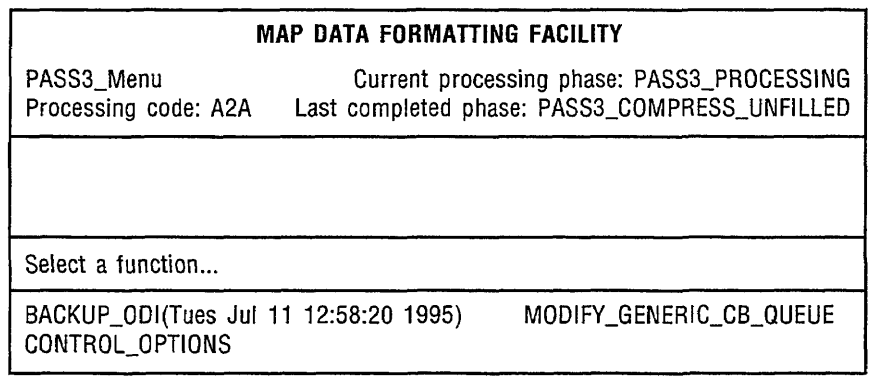



Fig. 73 — PASS3 menu at end of PASS3

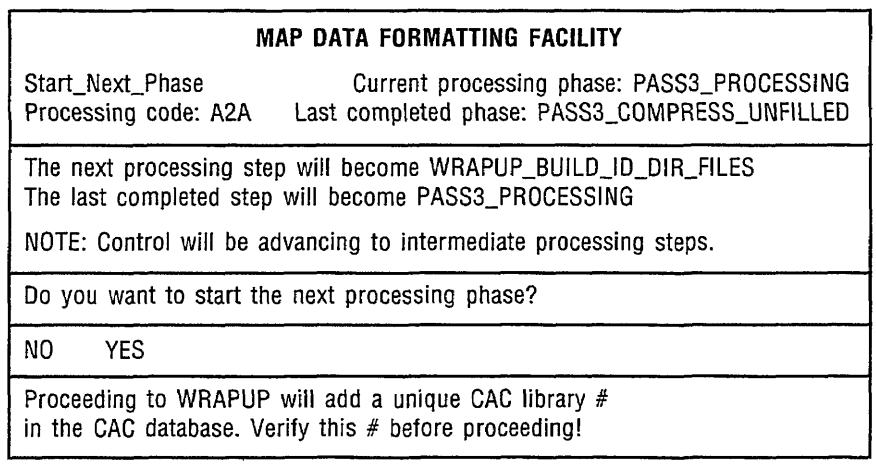



Fig. 74 — Advance to WRAPUP verification prompt

If this is a new CAC, a unique CAC library number will be assigned to the CAC ODI. If this is an update, the following rules will apply:

- The CAC will only be assigned a new CAC number if data have been *trimmed* from the original CAC boundaries; simply adding new data will not result in a new CAC number.
- If an ADRG CDROM (from an original CAC) is reprocessed in an update, the status_record reader field will contain the value REPROCESSED.

### 8.2 WRAPUP Steps and Menu

Once control is advanced to WRAPUP, the WAIT_STATE menu is provided since the first couple of steps in this phase are intermediate processing steps. Table 4 provides the status_record values from CHART_STATUS for each WRAPUP step. The first step is WRAPUP_BUILD_ID _DIR_FILES, which builds the AREADRC.DAT, AREASRC.DAT, CD_COVRG.DAT, and CD_ID.DAT files in the CHART_ODI_DISK:[ID] directory. Check these files for accuracy with the DUMP_CAC_FILES utility.

The second intermediate step is WRAPUP_CLEANUP, which deletes extraneous files, purges multiple versions of files, renames all files to version one (*.*;1), and verifies the existence of all essential files.

Chart_Control sends a mail message indicating whether errors occurred during WRAPUP_CLEANUP and creates the log file MDFF_SCRATCH:[000000]CLEAN_UP.LOG to list any errors that occurred and the projected CAC ODI size. Review this file if errors have occurred (note, file renaming is not an error during WRAPUP_CLEANUP). Correct any errors that were detected during *cleanup*, and then repeat the cleanup operation before advancing to the next phase. WRAPUP provides a menu with an option to perform cleanup again (Fig. 75). Once CLEANUP has been completed with no errors, advance control to the ISO image building phase.

Table 4 — Values of Status Record Control Fields ("Processing_Step" and "Last_Completed_Step") in CHART_STATUS.DAT File During WRAPUP. Values are Provided for Each ADRG CDROM Being Processed for a Given Thread.

| DESCRIPTION OF WRAPUP PROCESSING STEPS | PROCESSING_STEP | LAST_COMPLETED_STEP |
|---|---|---|
| **Building Header Files and Cleaning Up Directories** | | |
| Building ID directories and relevant files (Sec. 8.2) | WRAPUP_BUILD_ID_DIR_ FILES | PASS3_PROCESSING |
| Deleting extraneous files, purging old versions, etc. (Sec. 8.2) | WRAPUP_CLEANUP | WRAPUP_BUILD_ID_DIR_ FILES |
| Immediately following cleanup step (Sec. 8.2) | WRAPUP_PROCESSING | WRAPUP_CLEANUP |

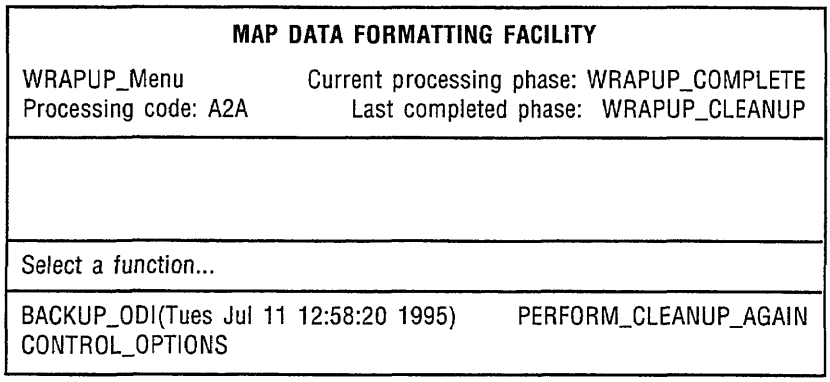



Fig. 75 — WRAPUP_PROCESSING menu

## 9.0 ISO IMAGE PROCESSING

The ISO processing phase builds an ISO 9660 formatted image of the CAC ODI and copies it to tape to be sent to the CDROM mastering facility.

This section describes preliminary considerations for ISO building and the steps that occur as the ISO image is built.

### 9.1 Preliminary Considerations

Adequate disk space must be available to accommodate the ISO 9660 formatted image. There must be at least 1.4 million free blocks available on the disk drive with the logical name ISO_IMAGES. Use the Diskuse command to determine available free disk space for this disk drive (Fig. 76).

If there is inadequate space on the current ISO_IMAGES disk drive, then find another disk drive with more than 1.4M free blocks and define that disk to ISO_IMAGES:

define/table = mdff ***ISO_IMAGES devicename:*[MDFF.IMAGES]**,

where *devicename* specifies the disk drive that contains adequate free disk space.

Now advance processing to the ISO Image processing phase by selecting CONTROL_OPTIONS from the WRAPUP menu. The START_NEXT_PHASE option (Fig. 77) advances to ISO image processing.

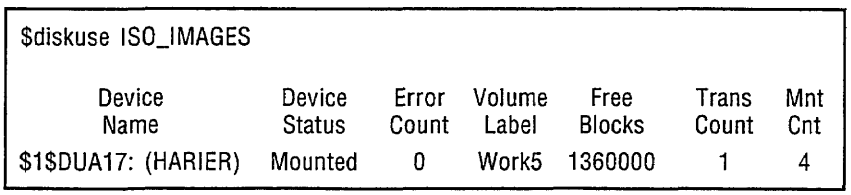



Fig. 76 — Disk space availability command and information

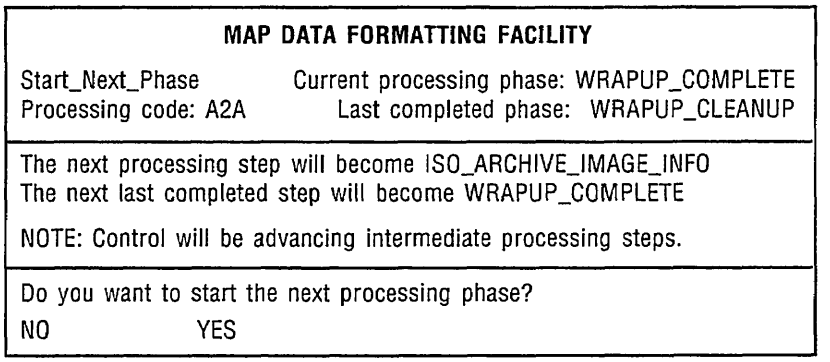



Fig. 77 — Start_Next_Phase menu at the end of WRAPUP

Table 5 — Values of Status Record Control Fields ("Processing_Step" and "Last_Completed_Step") in CHART_STATUS.DAT File During ISO BUILD. Values are Provided for each ADRG CDROM Being Processed for a Given Thread.

| DESCRIPTION OF ISO BUILD PROCESSING STEPS | PROCESSING_STEP | LAST_COMPLETED_STEP |
|---|---|---|
| **Building ISO-9660 Image File to Master onto CDROM** | | |
| Archiving information from CHART_STATUS.DAT (Sec. 9.2) | ISO_ARCHIVE_IMAGE_INFO | WRAPUP_PROCESSING |
| Building ISO-9660 image file (Sec. 9.2) | ISO_BUILD_IMAGE_FILE | ISO_ARCHIVE_IMAGE_INFO |
| After ISO-9660 image has completed (Sec. 9.2) | ISO_PROCESSING | ISO_BUILD_IMAGE_FILE |

## 9.2 ISO Image Build and Copy

Once control has advanced to the ISO image processing phase, the WAIT_STATE menu is provided since the first two steps in this phase are intermediate processing steps. Table 5 provides the status_record values from CHART_STATUS for each ISO_Image_Build step. The first step is ISO_ARCHIVE_IMAGE_INFO, which archives information from CHART_STATUS in the DATATRIEVE databases.

The second intermediate step is ISO_BUILD_IMAGE_FILE, which builds the CAC ISO image in the ISO_IMAGES directory. All CAC ISO image files follow a common file naming convention:

xxx_syyyyv.IMAGE,

where

xxx = the three-character processing thread name,
s = the one-character chart scale,
yyyy = the four-character ODI number,
v = the one-character CAC ODI version, and
IMAGE = the file extension.

For example, the CAC ISO image file named **A3C_30001C.IMAGE** is created for the A3C processing thread with TPC data (chart scale "3"), ODI number 1 (0001), version C.

After the ISO image has been built, Chart_Control displays an ISO_PROCESSING menu (Fig. 78). Before selecting either of the ISO_PROCESNG options (backup or copy), first verify that the ISO image has been built correctly.

Figure 79 is a sample of the files and directories for image A3C_30001C.IMAGE, which was generated by the CDDIR command:

*cddir/files* **ISO_IMAGES:A3C_30001C.IMAGE.**

**MAP DATA FORMATTING FACILITY**

COPY_ISO_IMAGE_MENU — Current processing phase: ISO_BUILD_COMPLETE
Processing code: A2A — Last completed phase: ISO_BUILD_IMAGE_FILE

Select an option...

BACKUP_ODI() COPY_ISO_IMAGE

Fig. 78 — ISO PROCESSING menu

```
Directory of ISO_IMAGE:[000000]
   file                 size     date        time      type
   ID                      4     9-28-1995   14:17:59  D
   MAP3                    8     9-28-1995   14:17:59  D
Total of       12 blocks in    2 files

Directory of ISO_IMAGE:[000000.ID]
   file                 size     date        time      type
   AREADRC.DAT;1           3     9-28-1995   14:17:59  F
   AREASORC.DAT;1          4     9-28-1995   14:17:59  F
   CD_COVRG.DAT;1          1     9-28-1995   14:17:59  F
   CD_ID.DAT;1             1     9-28-1995   14:17:59  F
Total of        9 blocks in    4 files

Directory of ISO_IMAGE:[000000.MAP3]
   file                 size     date        time      type
   CD200244                4     9-28-1995   14:17:59  D
   CD200747                4     9-28-1995   14:17:59  D
   CD200748                4     9-28-1995   14:17:59  D
   CD200749                4     9-28-1995   14:17:59  D
   CD200750                4     9-28-1995   14:17:59  D
   CD200751                4     9-28-1995   14:17:59  D
   CD200752                4     9-28-1995   14:17:59  D
```

Fig. 79 — Listing of A3C_30001C.IMAGE with Cddir/Files utility

After verification, use the COPY_ISO_IMAGE option (Fig. 78) to copy the CAC image to tape and send to the mastering facility. The BACKUP_ODI option creates a copy in the VMS BACKUP format, which is *not usable* by the mastering facility.

The COPY_ISO_IMAGE option submits a command procedure to MDFF_BATCH. After the job is completed, check the log file MDFF_SCRATCH:[000000]ISO_IMAGE_COPY_FILE.LOG for errors. The copy is considered usable when no errors have occurred. *Congratulations!* You have just successfully created a CAC CDROM.

## 10.0 REFERENCES

1. Myrick, S. A., P. B. Wischow, M. C. Lohrenz, M. E. Trenchard, M. L. Gendron, L. M. Reidlinger, and J. M. Mehaffey, "MDFF HELP Library, Version 2: On-Line Documentation for the Map Data Formatting Facility," NRL/MR/7441--93-7071, Naval Research Laboratory, Stennis Space Center, MS, 1994.

2. Myrick, S. A. and M. C. Lohrenz, "MDFF HELP Library Design Document," NRL/MR/7441--95-7682, Naval Research Laboratory, Stennis Space Center, MS, 1995.

3. Lohrenz, M. C. and J. E. Ryan, "The Navy Standard Compressed Aeronautical Chart Data Base," NOARL Report 8, Naval Research Laboratory, Stennis Space Center, MS, 1990.

4. Lohrenz, M. C., M. E. Trenchard, S. A. Myrick, P. B. Wischow, and L. M. Riedlinger, "The Navy Tessellated Spheroid Map Projection System: A Comprehensive Definition," NRL/FR/7441--92-9408, Naval Research Laboratory, Stennis Space Center, MS, 1993.

5. "The Military Specification for Arc Digitized Raster Graphics (ADRG)," DoD MIL-A-89007, provides a complete description of ADRG data, the source for CAC.

6. Lohrenz, M. C., "Military Specification the Navy Standard Compressed Aeronautical Chart Database," SP 024:351:91, Naval Research Laboratory, Stennis Space Center, MS, 1991.

# Appendix A

## GLOSSARY OF TERMS

*ADRG*

Equal Arc-Second Digitized Raster Graphics (ADRG) data that is distributed by the National Imagery and Mapping Agency (NIMA). This is the source data that is used to create the Compressed Aeronautical Chart (CAC) database.

*CDROM*

Compact Disk-Read Only Memory. Optical storage media that holds approximately 600 Mbytes. ADRG and CAC data are stored on CDROM.

*Column*

In the Tessellated Spheroid (TS) projection system, segments are arranged by latitudinally based rows and longitudinally based columns within the equatorial and temperate zones. The first column of segments in each nonpolar zone (column 0) is located with its western boundary on the 0° meridian. Positive columns extend eastward from column 0 and stop at the 180° meridian. Negative columns extend westward from column 0 and stop at the 180° meridian. Within a single CAC segment, columns of pixels are numbered from left (column 0) to right (column 255).

*Core Segment*

Segments that cover the innermost portions of a chart. Since core segments are typically filled with data, they are usually compressed in PASS1.

*Distribution Rectangle (DR)*

The bounding geographic coordinates of the data stored on an ADRG CDROM.

*Edge Segment*

Segments that contain portions of a chart's edges. It is also used to identify all partially filled TS segments for a given ADRG CDROM.

PASS2 fills each edge segment with all available ADRG data for that segment. Edge segments that are completely filled with data (usually from more than one chart) are referred to as *filled* edge segments; all other edge segments in the thread are referred to as *unfilled* edge segments. Unfilled edge segments are partially filled with data (they are not empty).

*Filled Segment*

A term used to identify previous edge segments from PASS1. PASS2 fills each edge segment with all available ADRG data for that segment. Edge segments that are completely filled with data (usually from more than one chart) are referred to as *filled* edge segments; all other edge segments in the thread are referred to as *unfilled* edge segments. Unfilled edge segments are partially filled with data (they are not empty).

*INITIALIZATION*

Defines a new processing thread and allocates system resources. During this phase, Chart_Control prompts the operator for the processing thread name, the chart scale of the data top be processed, and other relevant information.

*ISO_IMAGE_BUILD*

The final phase in CAC processing. The Optical Disk Image (ODI) is a "snapshot" of the entire file system (including directories and data files) to be mastered on CDROM. The mastering facility requires the ODI to be written in the ISO 9660 standard format. Once the ODI has been built in the ISO format, it is copied to tape and sent to the CDROM mastering facility.

*ISO Image*

The CAC Optical Disk Image (ODI) is written in the ISO 9660 standard format and mastered on CDROM. The mastering facility requires the ODI to be written in this standard CDROM format.

*Optical Disk Image (ODI)*

The ODI is a "snapshot" of the entire file system (including directories and data files) to be mastered on CDROM. The mastering facility requires the ODI to be written in the ISO 9660 standard format

*PASS1*

Begins the actual data processing by copying ADRG data from CDROM onto a local disk and compressing *core segments.* Core segments cover the innermost portions of a chart and *edge segments* contain portions of a chart's edges.

*PASS2*

Fills each *edge segment* with all available ADRG data for that segment. Edge segments that are completely filled with data (usually from more than one chart) are referred to as *filled* edge segments; all other edge segments in the thread are referred to as *unfilled* edge segments. Unfilled edge segments are partially filled with data (they are not empty). PASS2 also defines the exact area of coverage for the CAC Optical Disk Image (ODI). All segments that lie outside the defined area of coverage are *trimmed* before the ODI is built.

*PASS3*

Compresses the *filled* and *unfilled edge segments.* Hence, upon its completion, all of the data that are to be included in this CAC have been compressed.

*Processing Thread*

One processing thread is established for each new CAC to be processed. Establishing a processing thread involves allocating resources and establishing a unique file name for a given CAC. The status of any CAC-in-process can be precisely determined by examining the state of its processing thread.

A processing thread name is comprised of a three-character code: the first character distinguishes the data set type (for CAC, that must be an "A" for aeronautical chart); the second character identifies the scale of the data set, where (0)=1:50K, (1)=1:100K, (2)=1:250K, etc.; the third character is a sequence identifier (incremented from A to Z) for each new CAC at a given scale.

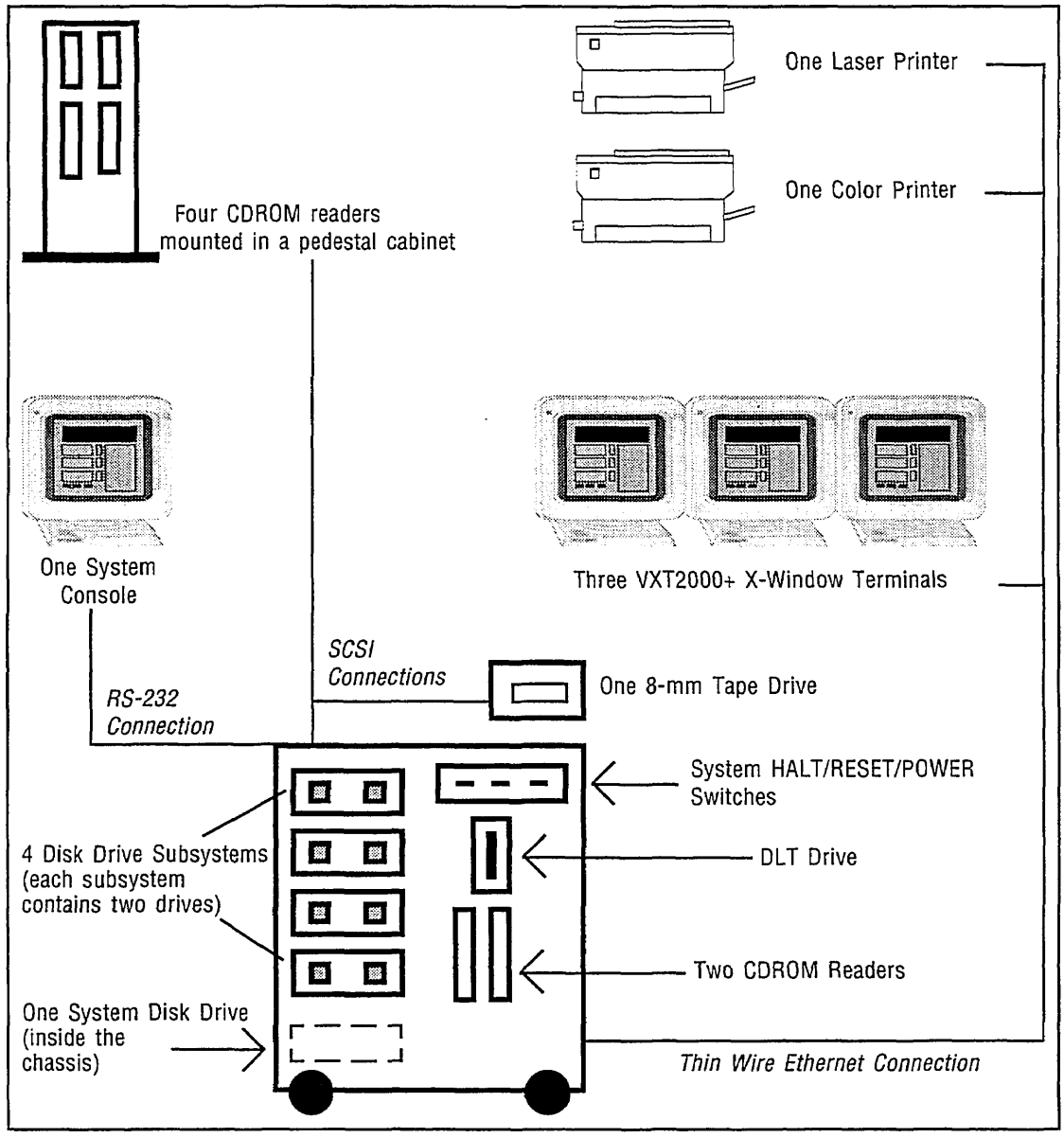



Fig. 5 — CPS hardware configuration

### 2.5.3 *Operator Console Messages*

Operator Console Messages (OPCOM) are used to monitor the CPS and relay all error and status messages (for additional information, type *help reply/enable*). The CPS sends notices to terminals that are enabled to receive OPER5 OPCOM notices and the system console receives all OPCOM notices by default. Figure 6 shows common commands for managing OPCOM information.

Much of the OPCOM text is written to the SYS$MANAGER:OPERATOR.LOG file. Since this file is often modified, many versions of this file are created. This file needs to be periodically purged to avoid using excessive disk space.

segments in the thread are referred to as *unfilled* edge segments. Unfilled edge segments are partially filled with data (they are not empty).

***WRAPUP***

Creates numerous CAC header files to be included in the Optical Disk Image (ODI), including files that store overall CAC CDROM coverage, color PA coverages, and area source and area DR information. WRAPUP also deletes any temporary "scratch" files and other extraneous files.

***Zone***

The TS projection system divides the Earth surface into five latitudinal *zones*; two zones that cover the polar regions–North Polar and South Polar, and three nonpolar zones–North Temperate, Equatorial, and South Temperate. Any use of this term refers to one on these TS zones.

## Appendix B

## UTILITY SOFTWARE

## CONTENTS

DISPLAY_ADRG_X* ..... 73

DISPLAY_SEGMENTS_X* ..... 80

DUMP_CAC_FILES ..... 90

LIST_CHART STATUS ..... 96

LIST_CODEBOOK STATUS ..... 99

PLOT_ODI* ..... 101

SQCODE ..... 110

SQMDFF ..... 111

THREAD ..... 112

CAC_QC* ..... 114

*Also available from network access

# DISPLAY_ADRG_X CONTENTS

B1.0 INTRODUCTION ..... 73
B1.1 Invocation ..... 73
B1.2 Viewing ADRG Data ..... 73
B1.2.1 Viewing the Overview Image ..... 74
B1.2.2 Viewing the Chart Image ..... 74
B1.3 Menu Bar Options ..... 76
B1.3.1 Palette Selection Option ..... 76
B1.3.2 Next Overview Option ..... 76
B1.3.3 Coordinate Entry Option ..... 76
B1.3.4 "Options" Option ..... 77
B1.3.5 Help Option ..... 77
B1.4 Mouse Buttons ..... 77
B1.4.1 Mouse Button1 ..... 77
B1.4.2 Mouse Button2 ..... 78
B1.4.3 Mouse Button3 ..... 78

# DISPLAY_ADRG_X

## B1.0 INTRODUCTION

Display_ADRG_X is used to view ADRG data from CDROM. Originally designed to aid in the selection of data for generating CAC color palettes, it is also used to check for data flaws that are discovered during CAC processing (e.g., unreadable tiles, missing data, and corrupted data).

There are two types of images on ADRG CDROMs: the digitized chart image (or images, when the chart transitions two or more ADRG ARC zones) and the overview image. The overview image is a 1/16th reduced image of the digitized chart data that enables the *total area of available coverage* to be more easily seen.

There can be multiple Distribution Rectangles (DR) on an ADRG CDROM. Each DR contains one overview image and may possess multiple Zone Distribution Rectangles (ZDR). Multiple ZDRs are simultaneously shown to provide a full screen display.

The first section of this appendix describes how to invoke Display_ADRG_X. The second section describes how to view chart and overview data, the third section describes data viewing techniques, and the fourth and fifth sections describe main menu options and mouse buttons.

### B1.1 Invocation

Prior to invocation, the ADRG CDROM must be physically loaded in a CDROM reader to be viewed:

*MOUNT_CDROM* **reader**, where **reader** = CDROM reader name.

Next, use the following command to invoke Display_ADRG_X:

*Display_ADRG_X* **reader**, where **reader** = the CDROM reader name.

The display window will appear with the main menu (Fig. B1).

### B1.2 Viewing ADRG Data

ADRG data are usually color-compressed before being displayed since most display devices are *pseudo-color* devices (i.e., only 256 colors are viewable as compared to 16 million colors for *true color* devices). The *Palette Selection* menu option (Sec. B1.3.1) is available to select the color

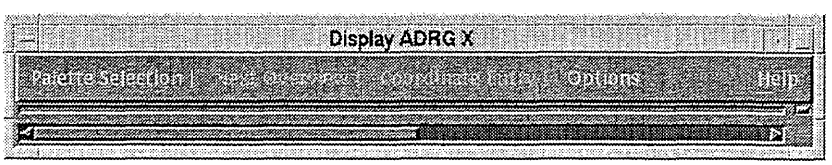



Fig. B1 — Main menu options

palette that is used for ADRG data color compression. The CAC color palette, which corresponds to the chart scale and zone, and the algebraic color palette may be selected for use.

### *B1.2.1 Viewing the Overview Image*

After a color palette has been selected for ADRG data color compression (Sec. B1.3.1), the first overview image is displayed. The cursor changes from the "Star Trek Enterprise" symbol to a wrist watch symbol while the program reads information from the transmitter header and general information files. The time required to read this information depends upon the ZDR size and its associated tile map.

Each time a tile is read, the cursor changes from the wrist watch symbol to an arrow symbol. Once the entire image is loaded, it is displayed and the cursor returns to the Enterprise symbol, and the ADRG chart series, scale, and color palette name are displayed in the top middle of the screen (Fig. B2). If there is more than one DR on the CDROM, the current DR number is also displayed.

Section B1.3.2 describes how to display other DRs with the *Next Overview* menu option and Sec. B1.3.3 describes how to view chart data with user-supplied coordinates with the *Coordinate Entry* menu option. Section B1.4.1 describes how to select areas of the chart image to view from the overview image using mouse button1.

### *B1.2.2 Viewing the Chart Image*

There are two methods for displaying areas to view: through latitude/longitude *Coordinate Entry* (Sec. B1.3.3) and through the use of mouse buttons with the overview image (Sec. B1.4.1).

There can be multiple chart images (i.e., more than one ZDR) when multiple ARC zones are written in a single DR. However, only one chart image (i.e., one ZDR) is viewable at a time. Therefore, if an area to be displayed spans *more than one image*, the desired full area of coverage will be truncated.

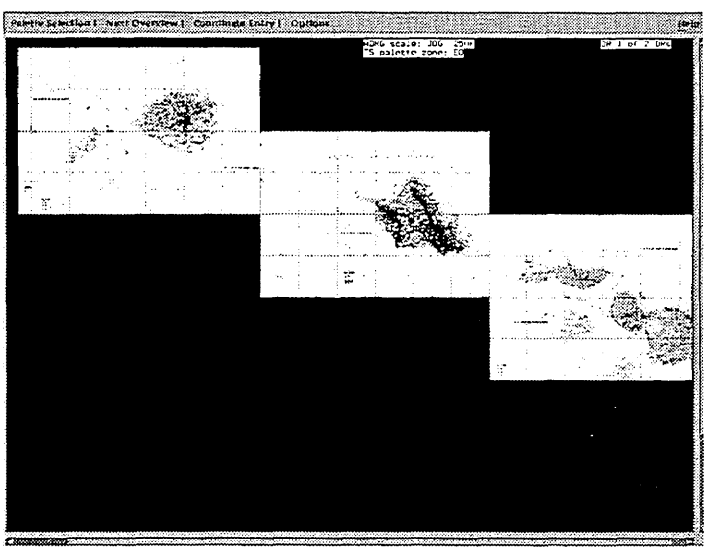


Fig. B2 — Display screen showing an overview image

To view the remaining area of coverage, select coordinates within that area and the previous area will be overwritten with new coverage. When an area of coverage is displayed, the current ZDR number and the total number of ZDRs on the CDROM is shown on the top of the screen (Fig. B3).

Chart image data can be viewed in two modes: ADRG mode (the default, Fig. B3) and downsampled mode (Fig. B4). Downsampled mode emulates the scale of CAC PASS1 ADRG downsampled data. Mouse button3 (Sec. B1.4.3) provides a toggle to switch between these two modes.

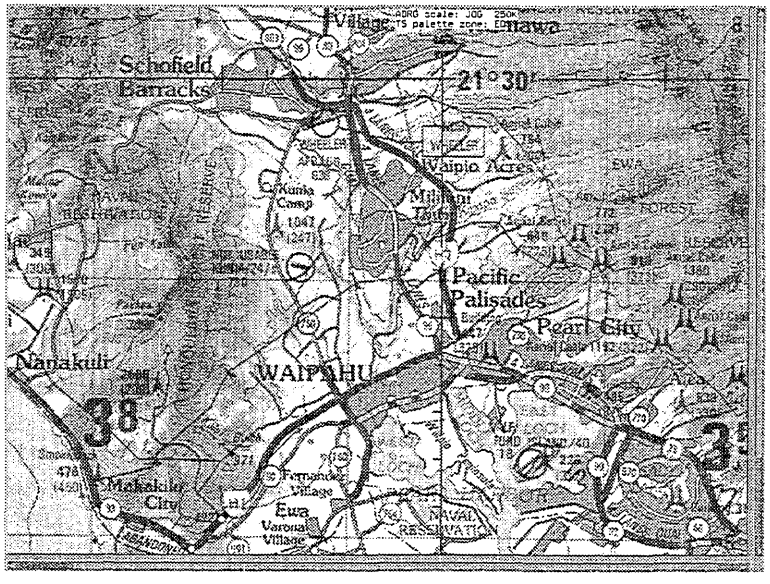



Fig. B3 — Chart image with current ZDR number located at the top of the display

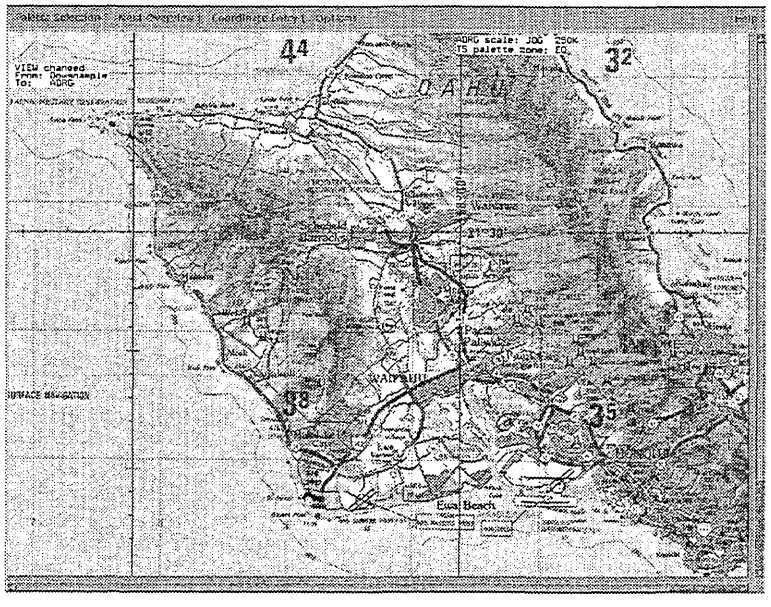



Fig. B4 — ADRG data displayed in downsampled mode

## B1.3 Menu Bar Options

Each of the main menu options (Fig. B1) are described within this section and include:

- PALETTE_SELECTION
- NEXT_OVERVIEW
- COORDINATE_ENTRY
- OPTIONS
- HELP

### *B1.3.1 Palette Selection Option*

This option is used to choose a palette to color compress ADRG data and must be selected before other options are available for use. All standard CAC color palettes and an algebraic color palette are offered. Note that coverages from chart scales and zones *without standard CAC palettes* can only use the algebraic palette. Selection of a palette automatically loads the overview image for the first (or in subsequent use, the next) DR.

After one palette has been chosen and used for display, another palette may be selected (if more are available). This selection loads the next overview image (or reloads the current image if there is only one DR) and displays it using the newly selected palette.

### *B1.3.2 Next Overview Option*

A palette must be selected prior to using this option (hence, when this option is initially unselectable (see Help Palette Selections)). There should be more than one DR on the CDROM (otherwise, this option will redisplay the same image).

DR overview images on a CDROM are viewed cyclically and in sequential order. Hence, after the last DR has been displayed, the first DR will be redisplayed. Single click mouse button2 (i.e., the middle mouse button, Sec. B1.4.2) to view ADRG data.

### *B1.3.3 Coordinate Entry Option*

A palette must be selected prior to using this option (hence, when this option is initially unselectable (see Palette Selection Option)). ADRG data can be viewed by either selecting a point with the cursor (see Help Mouse Button1) or by entering latitude and longitude coordinates. When selecting the Coordinate Entry option, latitude and longitude coordinates are entered by one of the following methods:

- Selecting the first button labeled "Degrees minutes seconds" and entering latitude and longitude coordinates as degrees, minutes, and seconds (DMS) of arc (25 15 00 latitude, –30 00 00 longitude).
- Selecting the second button labeled "Float Coords" (FC) and entering latitude and longitude coordinates as floating point (25.25 degrees latitude, –30.0° longitude) values.

When the DMS method is selected, floating point entry fields are inaccessible. To enter DMS coordinates:

- For latitude: place the cursor in the text box labeled "degrees" under "latitude" (or press the <TAB> key until the "degree" box is activated) and type the degree value (e.g., 25 for north

latitude). Next, tab over to the "minutes" box and enter the minutes value (e.g., 15), then tab over to the "seconds" box and enter the seconds value (e.g., 00).

- For longitude: Tab down to the "degrees" box under "longitude" and enter the longitude degrees, minutes, and seconds value (e.g., –30 00 00 for west longitude). Then tab down to the button labeled "display" and select it. Now the data will be retrieved from the CDROM, color-compressed, and displayed.

Based on these latitude/longitude coordinates, an ADRG tile will be displayed with the northwest corner of the tile located in the center of the display. To view another area, either reselect the DMS button and enter the new position or return to the overview image and select a point with mouse button1.

When the "FC" method is selected, a widget will pop up with the DMS entry fields rendered inaccessible. Tab over to the text entry field under the label "latitude" and enter a latitude value (e.g., 25.25). Next, tab over to the text labeled "longitude" and enter a longitude value (e.g., –30.0 for west longitude). Now, tab down to the "display" button and select it. Data will be read from ADRG CDROM, color-compressed, and displayed.

Note: Although a "cancel" button is also offered, no action is performed when it is selected.

### *B1.3.4 "Options" Option*

This option provides two buttons for selection: the *Quit* button, which exits the program and the *Cancel* button, which continues program execution and does not exit Display_ADRG_X.

### *B1.3.5 Help Option*

This option is used to display information about the menu buttons and functions of the mouse buttons (described in Sec. B1.4).

## B1.4 Mouse Buttons

### *B1.4.1 Mouse Button1*

*SINGLE CLICK:* After an overview image has been displayed, move the cursor to an area of interest and click mouse button1 once (left-most of the three buttons for a right-handed mouse). ADRG data are read from CDROM and displayed with the point selected located in the tile that has its northwest corner centered in the screen.

To display another area of ADRG data, single-click mouse button1 and the overview image will be redisplayed.

*DOUBLE CLICK:* No function is currently assigned for a double click of mouse button1.

*B1.4.2 Mouse Button2*

*SINGLE CLICK:* A single click of mouse button2 shows the latitude/longitude coordinates (in both DMS of arc and in floating point format) of the cursor position in the overview or ADRG image. This information is located near the upper-left corner of the image (Fig. B5).

*DOUBLE CLICK:* Double clicking mouse button2 removes the information that was displayed through the previous single click of mouse button2.

*B1.4.3 Mouse Button3*

*SINGLE CLICK:* A single click of mouse button3 toggles the main menu bar display. Initially it is displayed; a single click removes it, and the subsequent single click redisplays it.

*DOUBLE CLICK:* A double click of mouse button3 changes the view mode of ADRG data:

- ADRG mode, provides a pixel-to-pixel viewing of ADRG data.
- Downsample mode, provides a view of *every other* pixel in latitude and longitude. Pixels are not averaged.

A double click while displaying the overview image changes the viewing mode, but has no effect on the overview image itself, nor does it display ADRG data (see Help for Mouse Button1 to display ADRG data).

Double clicking while displaying ADRG data causes the program to retrieve data from CDROM and display it in the new viewing mode.

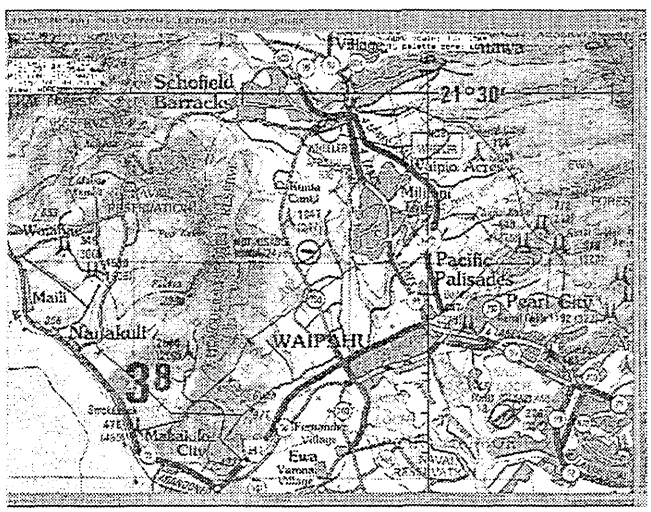



Fig. B5 — Latitude/longitude coordinate display

# DISPLAY_SEGMENTS_X CONTENTS

B2.0 INTRODUCTION ..... 80
B2.1 Invocation ..... 80
B2.2 Main Menu Options ..... 80
B2.2.1 File Option ..... 81
B2.2.1.1 Zone Button ..... 81
B2.2.1.2 ADRG Selections Button ..... 81
B2.2.1.3 Compressed Segments Button ..... 83
B2.2.1.4 Downsampled Segments Button ..... 84
B2.2.1.5 Segments File Button ..... 85
B2.2.1.6 Quit Button ..... 86
B2.2.2 World Option ..... 87
B2.2.3 Tools Option ..... 87
B2.2.3.1 The Zoom+ Button ..... 87
B2.2.3.2 The Zoom– Button ..... 88
B2.2.4 Print_Screen Option ..... 88
B2.3 Mouse Button3 ..... 88

# DISPLAY_SEGMENTS_X

## B2.0 INTRODUCTION

Display_Segments_X provides information about ADRG CDROMs that have been selected for processing, including a wire-frame diagram showing downsampled and compressed segments and an option for overlaying World Vector Shoreline (WVS) with political boundaries. WVS serves as a visual aid, for example, when a stray CDROM is accidentally included for processing. If CDROMs of *Central Europe* are being processed and coverage of North America is also included, this erroneous coverage will be noticeable!

Use of DISPLAY_SEGMENTS_X includes:

- Displaying the ADRG CDROMs that have been selected for processing via the data file ADRG_SELECTIONS.DAT (Sec. B2.2.1.2), which was produced by the PLOT_ODI program.
- Displaying downsampled and compressed segments on disk during PASS1 (Secs. B2.2.1.3 and B2.2.1.4).
- Validating the SEGMENTS.DAT file: During PASS1, each ADRG CDROM has a corresponding DS_SEGMENTS.DAT file that contains information about its downsampled segments. At the end of PASS1, all the DS_SEGMENTS.DAT files are combined and the segments sorted to generate a single file named SEGMENTS.DAT. DISPLAY_SEGMENTS_X should be used to confirm that this file has been properly built (Sec. B2.2.1.5).

DISPLAY_SEGMENTS_X is used to cross-check the SEGMENTS.DAT file against contents of the CHART_SEGMENTS and CHART_ODI_DISK disk drives:

- Only filled edge (PASS1 and PASS2 filled) and unfilled edge segments should be on the CHART_SEGMENTS disk.
- For compressed data, every segment on the CHART_ODI_DISK disk should be listed in the SEGMENTS.DAT file.

### B2.1 Invocation

Display_Segments_X is invoked using the following command:

*Display_Segments_X*, (where there is no input).

### B2.2 Main Menu Options

This section describes the main menu options (Fig. B6) that include: the *File* option (Sec. B2.2.1), the *World* option (Sec. B2.2.2), the *Tools* option (Sec. B2.2.3), and the *Print_Screen* button (Sec. B2.2.4).

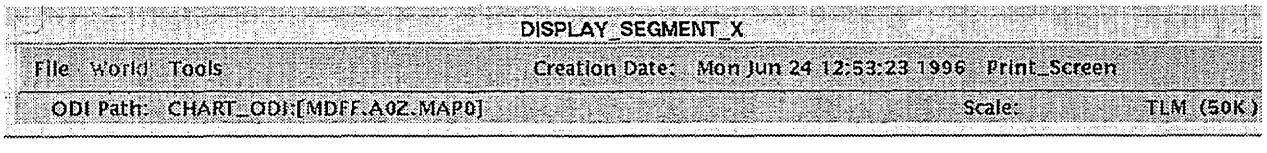


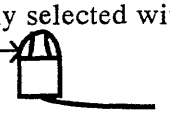

Fig. B6 — Main menu options

*B2.2.1 File Option*

This option provides the following buttons for selection (Fig. B7):

- ZONE (Sec. B2.2.1.1)
- ADRG SELECTIONS (Sec. B2.2.1.2)
- COMPRESSED SEGMENTS (Sec. B2.2.1.3)
- DOWNSAMPLED SEGMENTS (Sec. B2.2.1.4)
- SEGMENTS FILE (Sec. B2.2.1.5)
- QUIT (Sec. B2.2.1.6)

Program options are generally selected with a single click of mouse button1 (i.e., the left-most mouse button). CLICK

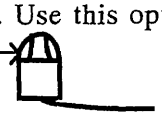


**B2.2.1.1 Zone Button**

Initially, the ZONE and QUIT buttons are the only selectable buttons. The Zone button is used to designate a TS zone and must be selected *before* any data can be displayed.

A single click of mouse button1 selects this button. Afterward, a popup window will be provided for zone selection (Fig. B8). CLICK

Move the cursor to the desired zone and double click mouse button1. The zone is now set and selection of other buttons is now possible. Use this option to select other TS zones and view data within these zones. CLICK CLICK



**B2.2.1.2 ADRG Selections Button**

This option displays the ADRG CDROMs from the PLOT_ODI file named ADRG_SELECTIONS.DAT. For nonpolar zones, all ADRG CDROMs in the selection file can be displayed. When a polar TS zone is selected, display is restricted to the bounding latitude boundaries. The result is a wire frame of the combined selected CDROMs with segments plotted in blue (Fig. B9). This data display is toggled on/off by clicking this button.

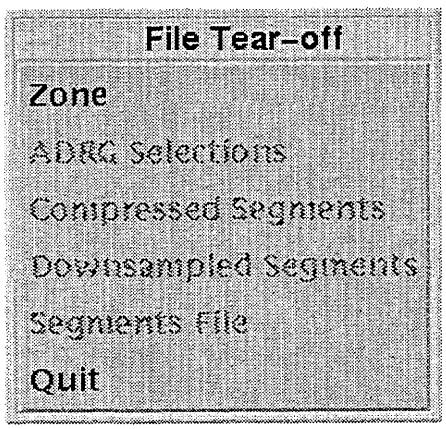



Fig. B7 — File menu options

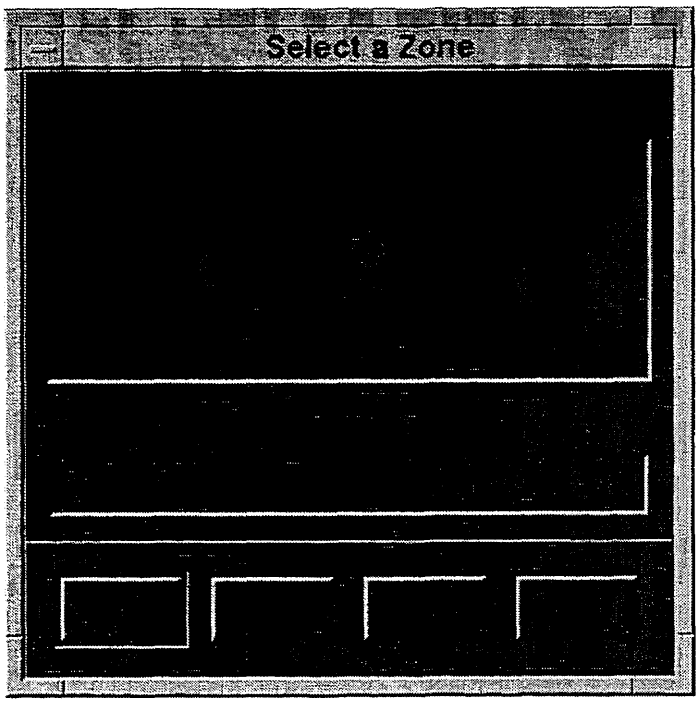



Fig. B8 — Zone button popup window

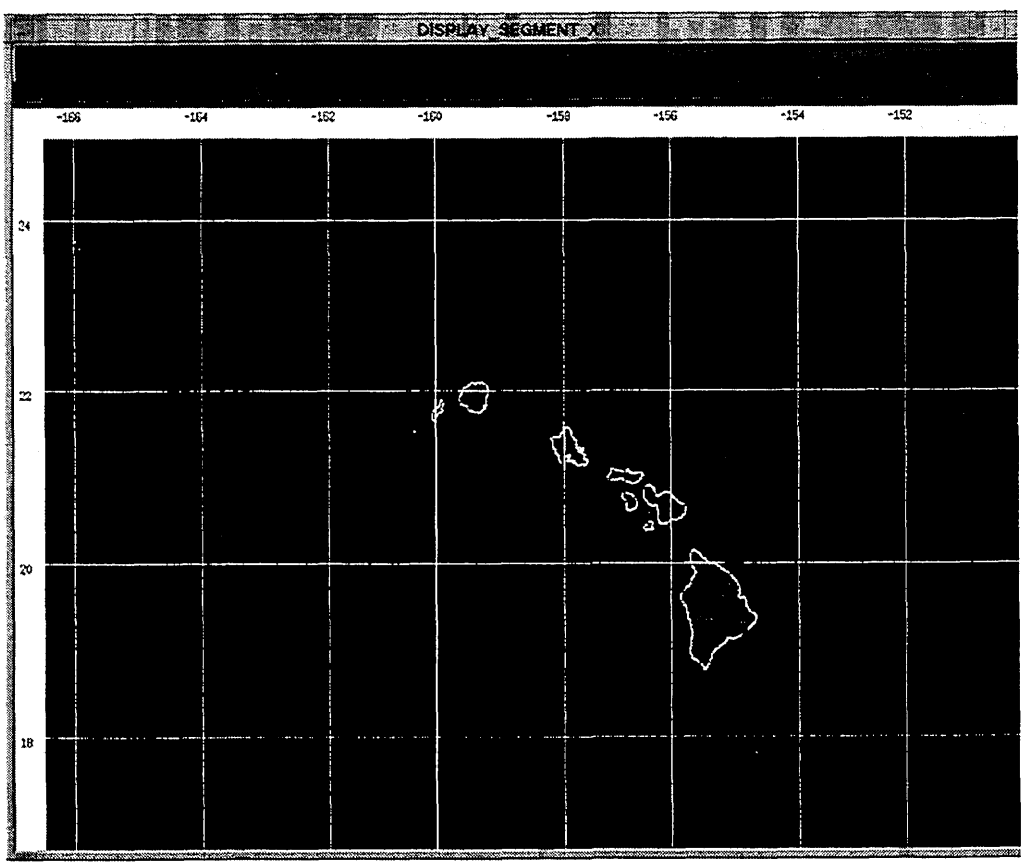



Fig. B9 — Selected ADRG CDROMs wire frame

### B2.2.1.3 Compressed Segments Button

This button displays all segments that are associated with the selected TS zone and stored on the CHART_ODI_DISK disk. These segments are plotted green in the wire frame. To view data from another zone, see Sec. B2.2.1.1.

This option is also used to monitor the progress of data compression (Fig. B10) as they are written to the CHART_ODI_DISK disk. Repeatedly selecting this button will show compression progressing (as more segments are stored). The reverse occurs with the downsampled data: as segments are compressed, the downsampled segments are deleted from the CHART_SEGMENTS disk (Sec. B2.2.1.4)!

After all ADRG CDROMs have completed PASS1 compression, the segments display on the CHART_ODI_DISK shows the outer edges of missing data. These are the edge filled and unfilled segments that will be compressed in PASS3A and PASS3B. Display of downsampled segments reveals that these segments exist, but are not yet compressed (Sec. B2.2.1.5).

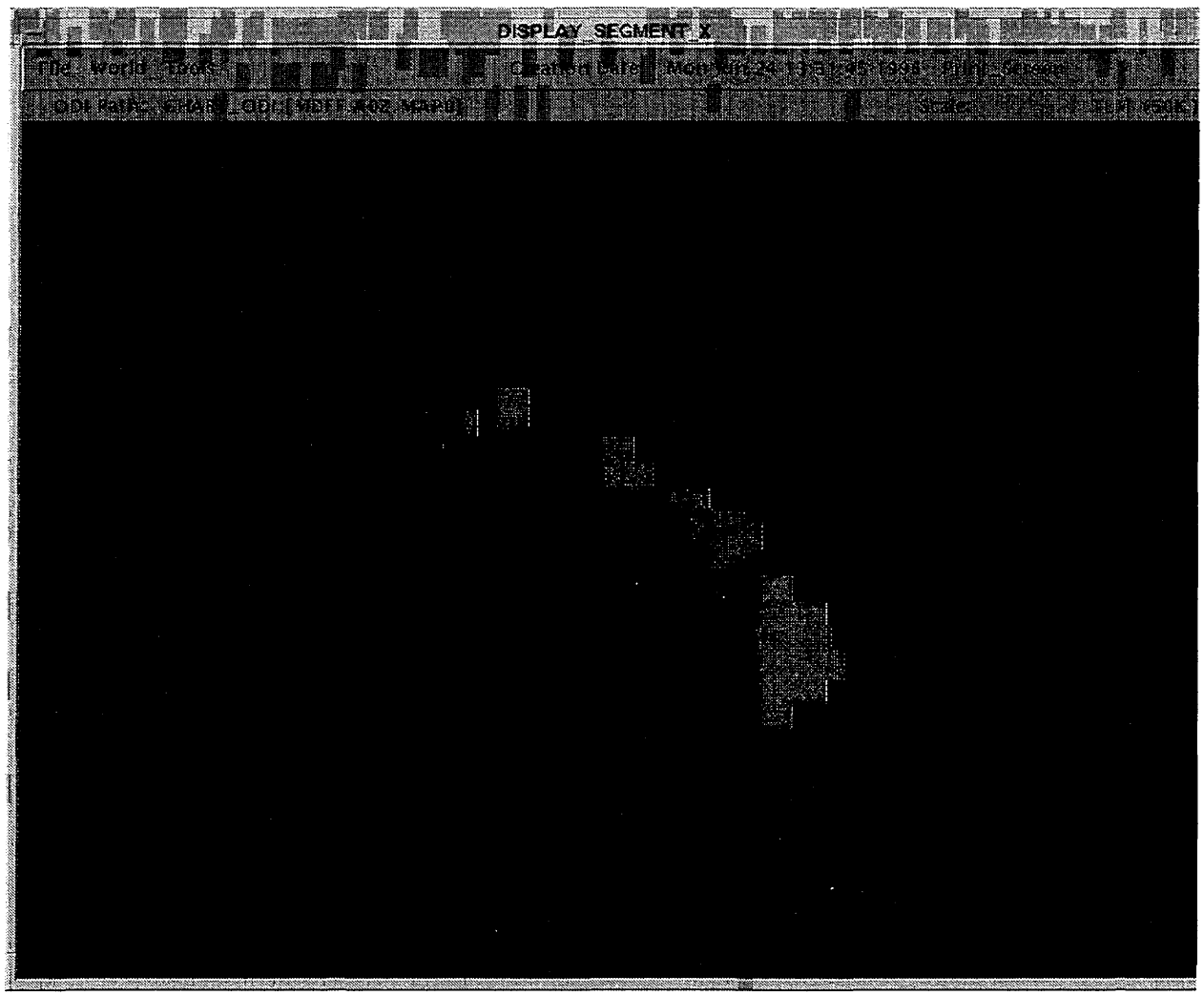



Fig. B10 — Compressed segments wire frame

A comparison with the ADRG selections display (Sec. B2.2.1.2) may not agree since coordinates for CDROM coverages are rectangular and *do not take into account* missing data and overlapping ADRG CDROMs (e.g., TPC G18 A, B, C, and R) as shown in Fig. B11.

### B2.2.1.4 Downsampled Segments Button

This button displays a wire frame of all segments stored on the CHART_SEGMENTS disk that are associated with the selected TS zone (Fig. B12). These segments are plotted in red. To view data from another zone, see Sec. B2.2.1.1.

Since downsample jobs store their segments on the CHART_SEGMENTS disk, their progression can be monitored by repeatedly selecting this button. The reverse can be observed for ADRG CDROMs that are being compressed; as segments are compressed, their corresponding downsampled segments are deleted from the CHART_SEGMENTS disk! After compression has been completed for an ADRG CDROM, only its unfilled segments will remain.

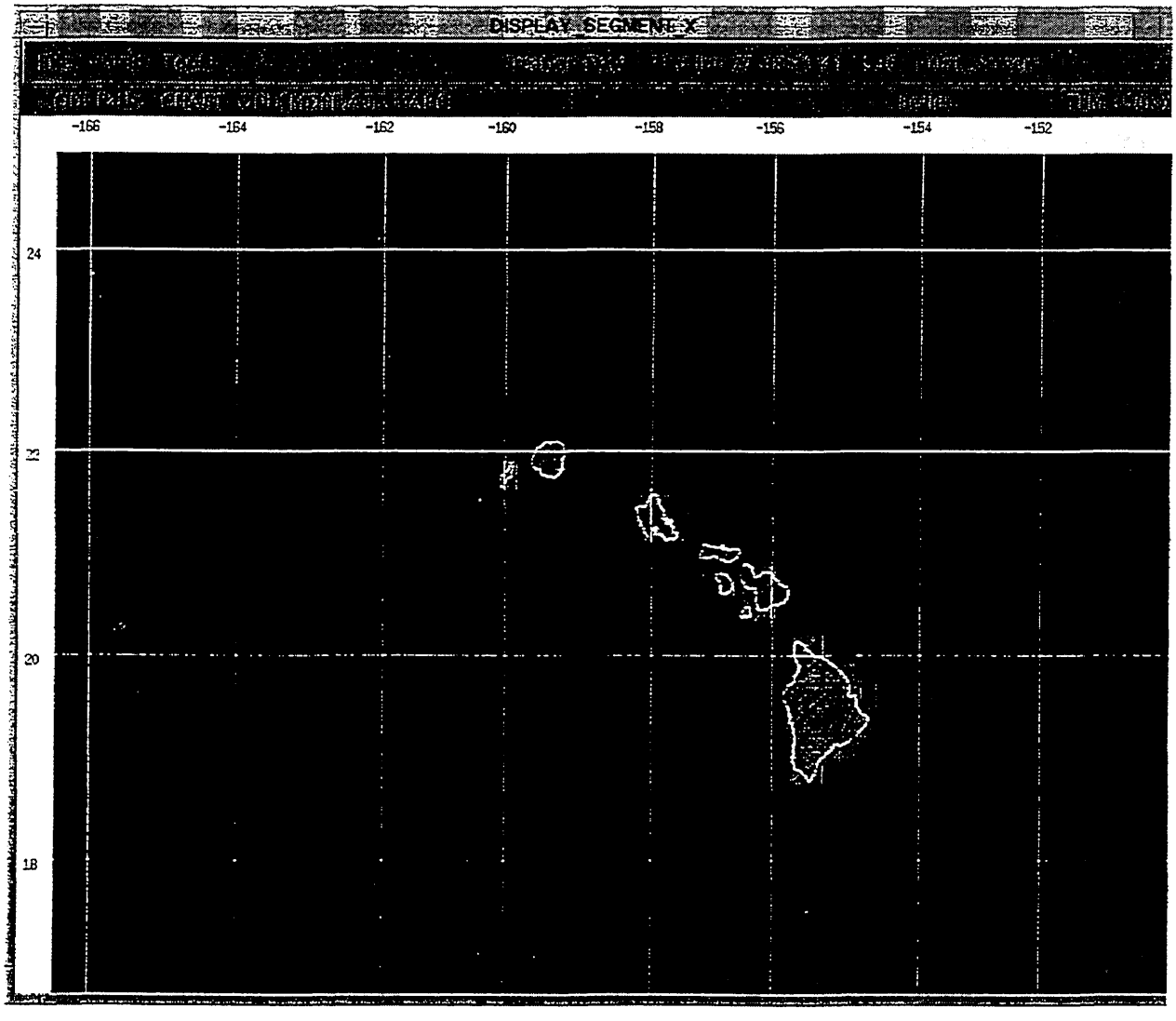



Fig. B11 — ADRG display showing actual areas of coverage and areas of missing data

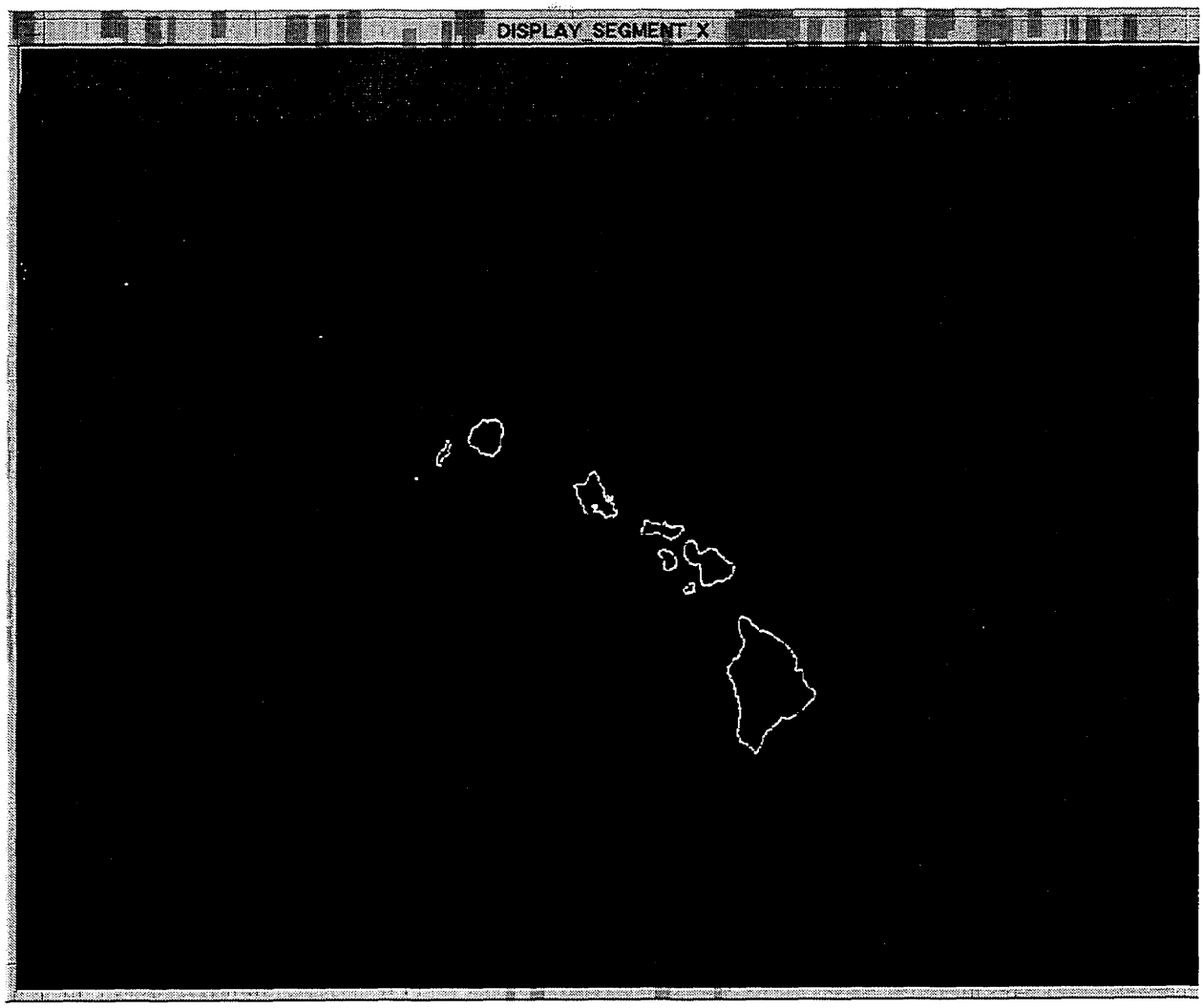



Fig. B12 — Downsampled segments wire frame

After all CDs have completed PASS1 compression, the wire-frame display of the data on the CHART_SEGMENTS disk should closely match the wire-frame display of the ADRG sections file (Sec. B2.2.1.2). However, they may not exactly match since the coordinates for ADRG CDROM coverages are rectangular and do not take into account missing data and overlapping ADRG CDROMs (i.e., TPC G18 A, B, C, and R).

### B2.2.1.5 Segments File Button

The SEGMENTS.DAT file is built at the end of PASS1 and contains all of the information from the DS_SEGMENTS.DAT files. Segments compressed in PASS1 are marked as being either "PASS1 filled" or "PASS1 compressed." Segments that are common to more than one ADRG CDROM and have a combined pixel count of 65,536 are marked "PASS1 filled" and will be compressed in PASS3A. Segments that are unfilled at the end of PASS1, but are filled in PASS2, will be marked "PASS2 filled" and compressed in PASS3A. Any remaining downsampled segments are marked "unfilled" and will be compressed in PASS3B.

This button will plot *any type* of downsampled or compressed segment in the SEGMENTS.DAT file. A popup selection box (Fig. B13) lists the possible combinations.

Double click mouse button1 to select the segments that will be displayed. Now another popup selection box will appear. This box controls whether trimmed data (if any) are to be displayed. Select the *TRIMMED SEGMENTS* option to have trimmed data displayed or the *NO TRIM SEGMENTS* option not to display any trimmed data. To be sure that the trim operation was performed correctly, a view with *TRIMMED DATA* must be done at least once!

#### B2.2.1.6 Quit Button

Select this button to exit the program.

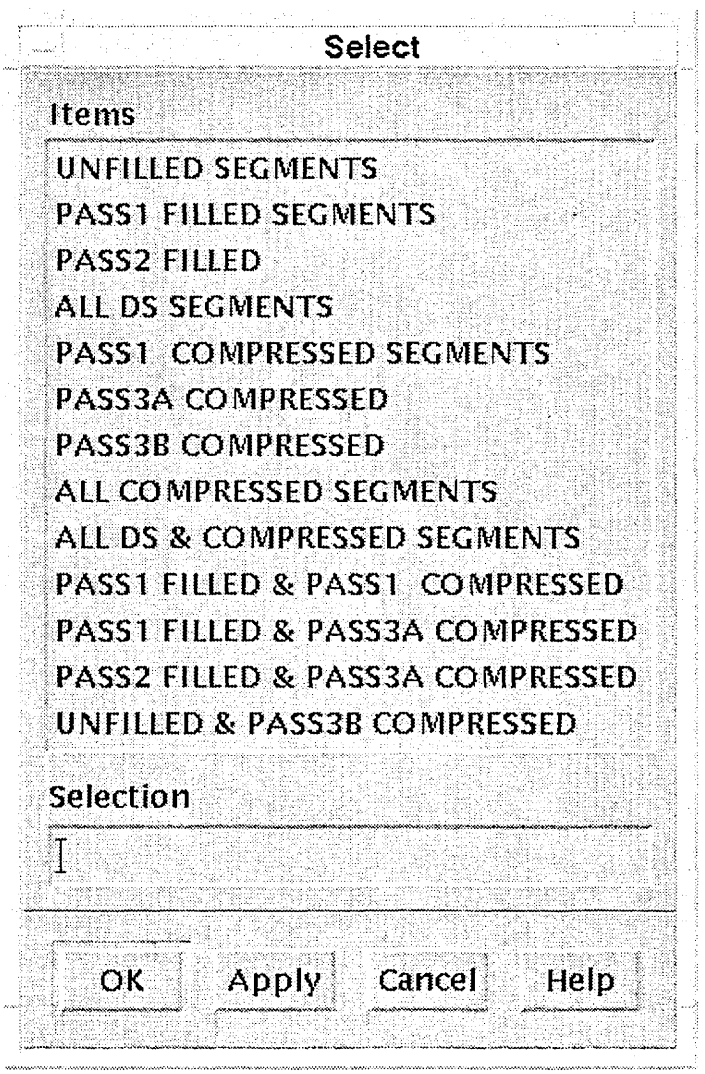



Fig. B13 — Segments file button popup selection box

*B2.2.2 World Option*

This option offers the following cascade buttons:

| | |
|---|---|
| *Refresh Display* | Refreshes the WVS overlay. |
| *Map* | Toggles the WVS display. The first selection of this button displays WVS, a second selection removes WVS. |
| *Lat/Lon Grid* | Toggles the latitude/longitude grid display. The option to change the latitude/longitude grid spacing is offered each time the grid is selected. |
| *Zoom Factor 1* | Returns to the original display size (Sec. B2.2.3). |

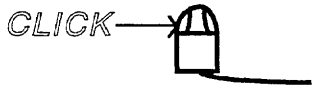
*B2.2.3 Tools Option*

This option provides display zoom functions. For smaller ADRG scales, individual segments appear rather small and hard to distinguish. The zoom feature allows the segments to be displayed more clearly.

To use these functions, first select the *ADRG SELECTIONS* button to obtain the desired area of coverage, then the tools option to select the desired zoom factor (Fig. B14). Note that zooming can be slow when a lot of information is to be displayed. Data (i.e., WVS, downsampled, or compressed) are displayed using the selected zoom factor. To return to the original display size, select the *WORLD* button and then *ZOOM FACTOR 1* button.

**B2.2.3.1 Zoom+ Button**

This button puts the function of mouse button1 in *zoom-up* mode. To zoom-up an area, select the ZOOM1 button. Next, place the cursor on the area that is to become the center of the display when zoomed and single-click mouse button1.



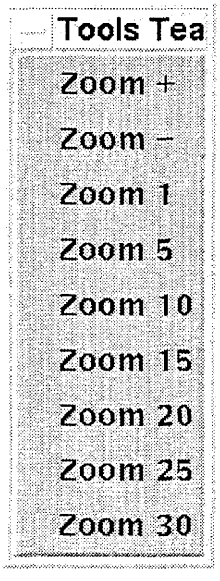




Fig. B14 — Tools option popup menu with zoom factors

Each successive click of mouse button1 will zoom in again, with current cursor position placed in the center (after zooming).

#### B2.2.3.2 Zoom– Button

This button puts the function of mouse button1 in a *zoom-down* mode. Once selected, each click of mouse button1 will zoom down the image.

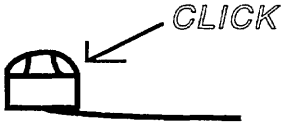

Subsequent mouse button1 clicks function similarly until a zoom factor of one is reached, after which zooming down has no further effect.

NOTE: If the intent of selecting "ZOOM–" is to return to a zoom factor of one, use the "ZOOM FACTOR 1" button (Sec B2.2.3)!

### *B2.2.4 Print_Screen Option*

Selecting this button will print a copy of the display on the TEKTRONIX printer. Since they may be used as future references (such as UPDATES), use it to create the following printouts for the processing logbook: ADRG CDROM selections, end of PASS1, end of PASS2, and end of PASS3 (prior to WRAP_UP).

### B2.3 Mouse Button3




Mouse button3 (i.e., the right-most button on a mouse) is used to obtain information about a segment on the display. Move the cursor to a segment on the display and single-click mouse button3. A window appears on the upper left of the display containing the latitude/longitude coordinates of the pixel that the cursor lies upon, the row and column of the segment, the latitude/longitude coordinates of the lower-left (southwest) corner of the segment, and the segment key name. Information is not available for segments outside of the selected TS zone.

# DUMP_CAC_FILES CONTENTS

B3.0 INTRODUCTION ..... 90
B3.1 Option1: AREASORCE.DAT File ..... 90
B3.2 Option2: AREACOVRG.DAT File ..... 91
B3.3 Option3: CD_COVRG.DAT File ..... 91
B3.4 Option4: COVERAGE.DAT File ..... 91
B3.5 Option5: CDHEADER.DAT File ..... 92
B3.6 Option6: Distribution Header File ..... 93
B3.7 Option7: Distribution Header File ..... 93
B3.8 ADRG CDROM Selection by TS Zone ..... 94

# DUMP_CAC_FILES

## B3.0 INTRODUCTION

DUMP_CAC_FILES lists the contents of .DAT files built during the WRAP_UP phase of CAC processing. It can also list these files from a CAC CDROM. The appropriate thread name must be set *prior to* invocation:

*DUMP_CAC_FILES* **device name**, where device name = device assigned the logical name CHART_ODI_DISK.

To list files on a CAC CDROM, specify the CDROM reader name (e.g., cdrom06) on which the CAC CDROM has been mounted (Fig. B15).

After the device name has been entered, the file named [ID] CD_ID.DAT is read and the CAC identification number is displayed (Fig. B15). The CAC identification number is translated as shown in Fig. B16, and should be validated.

Next, a list of *selection options* is displayed (Fig. B17). For quality control purposes, select and review information from each option, making sure that the output is valid. The following sections of this appendix describe each option.

Note: **Option 9** is used to exit the program, even though it is not part of the selection list.

## B3.1 Option1: AREASORCE.DAT File

This option lists the contents of the AREASORC.DAT file (Fig. B18). This file contains a list of path names of the source graphics header files (see option 7, SG HEADER.DAT) for each chart contained on ADRG CDROM. TPC scale charts typically contain only one chart per ADRG CDROM and, therefore, have one path name. JOG scale charts may contain from one to six charts per ADRG CDROM and, as a result, there will be a path name for each chart. If any of the data on a given

```
$ DUMP_CAC_FILES
Device name (CTRL_Z to quit): cdrom6
CAC ID: CD-1993-A-MAP2-00022
```

Fig. B15 — Example of program invocation listing a CAC identification number obtained from CDROM

```
CD-1993-A-MAP2-00022
        | |  |  | | |
        1 2  3  4 5 6
    1-Year CD was mastered
    2-Edtion (A=1, B=2,...)
    3-Indicates Aeronautical charts
    4-Scale (0=TLM,1=TCM,2=JOG,3=TPC,4=ONC,5=JNC)
    5-Mastering facility flag (0=3M, 1=Discovery Systems)
    6-CAC sequence number
```

Fig. B16 — CAC identification number translation

```
(1) AREASORC.DAT
(2) AREADRC.DAT
(3) CD_COVRG.DAT
(4) COVERAGE.DAT
(5) CD HEADER.DAT
(6) DR HEADER.DAT
(7) SG HEADER.DAT
(8) ADRGs by TS zone

Enter dump selection:
```

Fig. B17 — Option selection list

```
Enter dump selection: 1
Number of map areas: 2

Map area 1 subdirectory name: PA012502
Number of Source Graphics used for this area: 12

Source graphics path names:
CD000355\JAUS01\S01HED
CD000355\JAUS01\S02HED
.
.
Map area 2   subdirectory name: PA012601
Number of Source Graphics used for this area: 84

Source graphics path names:
CD000354\JAUS01\S01HED
CD000354\JAUS01\S02HED
.
.
```

Fig. B18 — AREASORCE.DAT file listing

ADRG CDROM spans more than one TS zone, then the path name(s) will be repeated in each TS zone.

### B3.2 Option 2: AREACOVRG.DAT File

This option lists the contents of the AREADRC.DAT file (Fig. B19), which contains the path name for the CD HEADER.DAT file (option 5, CD HEADER.DAT). The path name is repeated for each map area (i.e., TS zone) in which an ADRG CDROM contained data.

### B3.3 Option 3: CD_COVRG.DAT File

This option lists the contents of the CD_COVERAGE.DAT file (Fig. B20), which contains the minimum and maximum latitude/longitude coordinates of data that are contained within each TS zone on the given CAC CDROM.

### B3.4 Option 4: COVERAGE.DAT File

This option lists the COVERAGE.DAT file (Fig. B21), which contains the minimum/maximum latitude/longitude coordinate coverage of a color palette for a TS zone.

```
Enter dump selection: 2
Number of map areas: 2

Map area 1 subdirectory name: PA012502
Number of Source Graphics used for this area: 3

Source graphics path names:
CD000355\JAUS01
CD000359\JAUS01
CD000502\JAUS01

Map area 2 subdirectory name: PA012601
Number of Source Graphics used for this area: 23

Source graphics path names:
CD000354\JAUS01
CD000355\JAUS01
.
.
```

Fig. B19 — AREADRC.DAT file listing

```
Enter dump selection: 3
PANAME: PA012502  ZONE: 3

West Lon:   -92.07236481   (deg: -92 min:  4 sec: 20.513306)
East Lon:   -79.93421173   (deg: -79 min: 56 sec:  3.162231)
South Lat:   31.15384674   (deg:  31 min:  9 sec: 13.848267)
North Lat:   32.07692337   (deg:  32 min:  4 sec: 36.924133)

PANAME: PA012601  ZONE: 2

West Lon:   -92.01316071   (deg: -92 min:  0 sec: 47.378540)
East Lon:   -60.39473724   (deg: -60 min: 23 sec: 41.054077)
South Lat:   11.88461494   (deg:  11 min: 53 sec:  4.613800)
North Lat:   31.61538506   (deg:  31 min: 36 sec: 55.386200)
```

Fig. B20 — CD_COVRG.DAT file listing

```
Enter dump selection: 4
Palette coverage of PA012502

West Lon:   -180.00000000  (deg: -180 min:  0 sec:  0.000000)
East Lon:    180.00000000  (deg:  180 min:  0 sec:  0.000000)
South Lat:   31.38461494   (deg:   31 min: 23 sec:  4.613800)
North Lat:   51.69230652   (deg:   51 min: 41 sec: 32.303467)

Palette coverage of PA012601

West Lon:   -180.00000000  (deg: -180 min:  0 sec:  0.000000)
East Lon:    180.00000000  (deg:  180 min:  0 sec:  0.000000)
South Lat:  -31.38461494   (deg:  -31 min: 23 sec:  4.613800)
North Lat:   31.38461494   (deg:   31 min: 23 sec:  4.613800)
```

Fig. B21 — COVERAGE.DAT File listing

### B3.5 Option 5: CDHEADER.DAT FILE

This option lists the contents of the CDHEADER.DAT file. The MDFF sequence number of the CDROM is supplied as input (Fig. B22). Enter ^Z at the CDROM prompt to return to the main menu.

```
Enter dump selection: 5
CD# (CNTL-Z to return to menu): 355
TRANSMITTAL_DESCRIPTION_RECORD
        transmitter text
CD# (CNTL-Z to return to menu): ^Z
```

Fig. B22 — CDHEADER.DAT file listing

### B3.6 Option 6: DISTRIBUTION HEADER FILE

This option lists information contained in the DR file. Enter the MDFF sequence number of the ADRG CDROM whose DR will be listed (Fig. B23). Enter ^Z at the CDROM prompt to return to the main menu.

### B3.7 Option 7: DISTRIBUTION HEADER FILE

This option lists source graphics header files. All of the graphic header files for the specified ADRG CDROM will be listed (Fig. B24). If data were processed for more than one TS zone, then files will be listed for each TS zone.

```
Enter dump selection: 6
CD# (CNTL-Z to return to menu): 355
DATA_SET_DESCRIPTION_RECORD
.
. data set text is displayed here
.
CD# (CNTL-Z to return to menu): ^Z
```

Fig. B23 — Distribution header file listing

```
Enter dump selection: 7
CD# (CNTL-Z to return to menu): 355
*** PA directory: PA012502  ***
SG file: cdrom6:[MAP2.CD000355.JAUS01]S01HED.DAT
.
.
*** PA directory: PA012502  ***
SG file: cdrom6:[MAP2.CD000355.JAUS01]S02HED.DAT
.
.
*** PA directory: PA012601 ***
SG file: cdrom6:[MAP2.CD000355.JAUS01]S01HED.DAT
.
.
*** PA directory: PA012601 ***
SG file: cdrom6:[MAP2.CD000355.JAUS01]S02HED.DAT
.
.
CD# (CNTL-Z to return to menu): ^Z
```

Fig. B24 — Source graphics header files

## B3.8 ADRG CDROM SELECTION BY TS ZONE

Selection of option 8 lists the stock numbers of all the NIMA ADRG CDROMs that were processed as part of this CAC (Fig. B25).

Enter dump selection: 8

Zone: NT

| ADRG Sequence # | ADRG Stock # | Edition | Pub Date |
|---|---|---|---|
| 00355 | ARC1 1501A052L | 1 | 1989 06 14 |
| 00359 | ARC1 1501A052K | 1 | 1989 06 02 |
| 00502 | ARC1 1501A052J | 1 | 1989 06 01 |

Zone: EQ

| ADRG Sequence # | ADRG Stock # | Edition | Pub Date |
|---|---|---|---|
| 00354 | ARC1 1501A052N | 1 | 1989 06 19 |
| 00355 | ARC1 1501A052L | 1 | 1989 06 14 |
| 00356 | ARC1 1501A052P | 1 | 1989 06 01 |

.
.

Fig. B25—ADRG CDROM stock number listing

# LIST_CHART_STATUS CONTENTS

B4.0 INTRODUCTION AND INVOCATION ..... 96
B4.1 All Qualifier ..... 96
B4.2 CD_Number Qualifier ..... 96
B4.3 Entry Qualifier ..... 96
B4.4 Full Qualifier ..... 96
B4.5 Header Qualifier ..... 96
B4.6 Output Qualifier ..... 97
B4.7 Pass Qualifier ..... 97
B4.8 Print Qualifier ..... 97
B4.9 Copies Qualifier ..... 97
B4.10 Stock_Number Qualifier ..... 97

# LIST_CHART_STATUS

## B4.0 INTRODUCTION AND INVOCATION

LIST_CHART_STATUS lists information about the CHART_STATUS file of the current or specified processing thread. LIST_CHART_STATUS is invoked through the following command, which lists all CHART_STATUS entries for the current processing thread:

*List* [**processing thread name**] [**qualifier(s)**].

If a processing thread has not been defined, a prompt for a valid thread name is issued. The following example lists all CHART_STATUS entries for processing thread A3B:

*List* **A3B.**

### B4.1 All Qualifier

Lists every CHART_STATUS entry:

*List /All.*

### B4.2 CD_NUMBER Qualifier

Lists CHART_STATUS entries that possess the specified MDFF CDROM number(s):

*List /CD_Number* = (**CD number**)[,...]).

The following example lists CHART_STATUS entries with MDFF CD numbers 511 and 600:

*List /CD_Number* = (**511,600**).

### B4.3 Entry Qualifier

Lists specified CHART_STATUS entry number(s):

*List /Entry* = (**entry number**)[,...]).

### B4.4 Full Qualifier

Lists the complete status record (the default value is NOFULL):

*List /[No]Full* (default).

### B4.5 Header Qualifier

Includes the header record as part of the listing (the default is NO HEADER):

*List /[No]Header.*

The following example lists only the header record and uses the reserved word ***Only***:

*List /Header* = ***Only***.

### B4.6 Output Qualifier

Directs the output to the specified file. By default, output is sent to SYS$OUTPUT:

*List /Output* **[= file specification]**.

The following command lists all processing thread A3B entries to an output file named MYFILE.OUT and sends a copy to the print queue named HARIER$LZ1:

*List* A3B */Print* = **HARIER$LZ1** */Output* = **MYFILE.OUT**.

### B4.7 Pass Qualifier

Lists every CHART_STATUS entry associated with the specified processing phase. By default, only entries associated with PASS1 are listed:

*List /Pass* **[= pass number]**.

### B4.8 Print Qualifier

Prints a listing on the specified printer. By default, output is sent to the print queue defined by the DCL symbol MDFF_PRINT_QUEUE:

*List /Print* **[= queue name]**.

The following example lists all processing thread A3B entries to an output file named MYFILE.OUT and sends it to the print queue named HARIER$LZ1:

*List* **A3B** */Print* = **HARIER$LZ1** */Output* = **MYFILE.OUT**.

### B4.9 Copies Qualifier

Indicates the number of copies to be printed and is used with the printer qualifier:

*List /Print /Copies* = #,

where # is the number of copies to be printed on SYS$OUTPUT.

### B4.10 Stock_Number Qualifier

Lists CHART_STATUS entries that contain the specified NIMA stock number(s):

*List /Stock_Number* = **(stock number)**[,...]).

The following example lists all CHART_STATUS entries with the substring "TPCXX" as part of their NIMA Stock Number. The header record is also displayed:

*List* **A3B** */Stock_Number* = **TPCXX** */Header*.

## LIST_CODEBOOK_STATUS CONTENTS

B5.0 INTRODUCTION AND INVOCATION ..... 99
B5.1 Output Qualifier ..... 99
B5.2 Print Qualifier ..... 99

# LIST_CODEBOOK_STATUS

## 5.0 INTRODUCTION AND INVOCATION

This utility lists the contents of a CODEBOOK_STATUS.DAT file. An *MDFF sequence number* is required with each invocation and is specified with the /CD_Number qualifier:

*List_Codebook_Status /CD_Number* = **#**, where **#** = an *MDFF sequence number.*

The following example lists the CODEBOOK_STATUS file for the ADRG CDROM with sequence number 5001:

*List_Codebook_Status /CD_Number* = **5001**

CAC color palette codebooks are listed with the /PA_Number qualifier:

*List_Codebook_Status /Pa=**PA#***, where **PA#** is the palette ID number

The following example lists the CODEBOOK_STATUS file for color palette number 130 and ADRG CDROM with sequence number 5445:

*List_Codebook_Status /Pa=* **130** */CD_Number* = **5445**

### B5.1 Output Qualifier

Directs output to the specified file. By default, output is sent to sys$output:

*List_Codebook_Status /Output=**[file specification]***

The following command lists information about ADRG CDROM with sequence number 5445 to the file named MYFILE.DAT:

*List_Codebook_Status /CD_Number =5445 /Output=**[file specification]***

### B5.2 Print Qualifier

Prints a listing on the specified printer. By default, output is sent to the print queue defined by the DCL symbol MDFF_PRINT_QUEUE:

*List_Codebook_Status /Print=**[queue name]***, where **queue name** = a print queue name

The following command lists the CODEBOOK_STATUS file for ADRG CDROM with sequence number 5445 and sends it to the print queue named HARIER$LZ1:

*List_Codebook_Status /CD_Number =5445 /Print=**HARIER$LZ1***

# PLOT_ODI CONTENTS

B6.0 INTRODUCTION ..... 101
B6.1 Invocation ..... 101
B6.2 Program Usage ..... 101
B6.3 Menu Bar Options ..... 102
B6.3.1 File Option ..... 102
B6.3.1.1 Scale Button ..... 103
B6.3.1.2 Print Button ..... 104
B6.3.1.3 Quit Button ..... 104
B6.3.2 Selection Box Option ..... 104
B6.3.3 World Option ..... 104
B6.3.3.1 Refresh Display Button ..... 105
B6.3.3.2 Map Button ..... 105
B6.3.3.3 Lat/Lon Grid Button ..... 105
B6.3.3.4 Zoom Factor One Button ..... 105
B6.3.3.5 Toggle Legend Box Button ..... 105
B6.3.3.6 Display ADRGs Button ..... 105
B6.3.4 Tools Option ..... 105
B6.3.4.1 Zoom In Button ..... 106
B6.3.4.2 Zoom Out Button ..... 106
B6.3.5 Print_Screen Option ..... 107
B6.4. Mouse Button Functions ..... 107
B6.4.1 Mouse Button1: ADRG CDROM Selection ..... 108
B6.4.2 Mouse Button3: Coordinates ..... 108
B6.5 ADRG Selection File ..... 108
B6.5.1 Selection File Contents ..... 108
B6.5.2 ADRG Selection File Name and Location ..... 109
B6.5.2.1 Processing Thread Assigned ..... 109
B6.5.2.2 No Processing Thread Assigned ..... 109

# PLOT_ODI

## B6.0 INTRODUCTION

PLOT_ODI is used to determine which areas of coverage have been processed into CAC and as a planning tool for selecting ADRG CDROMs for CAC processing. Once a chart scale has been selected, all the ADRG CDROMs that are registered in the MDFF DATATRIEVE library (for the designated scale) are identified and their areas of coverage displayed. A WVS overlay is optionally available for use.

### B6.1 Invocation

PLOT_ODI is invoked using the following command:

*PLOT_ODI.*

If the program fails to execute, verify the DECnet/x-terminal address scheme:

- Obtain the DECnet address assigned to the x-terminal from the terminal manager window.
- Define the monitor in use as the current display device:

  *Set Display/Create/Node* = **x.xxx**, where **x.xxx** is the x-terminal DECnet address.

Try executing the program again. If the program still fails, notify the system manager.

### B6.2 Program Usage

This section provides information about PLOT_ODI usage: Section B6.3 describes the buttons on the Menu Bar, Sec. B6.4 describes mouse button functions, and Sec. B6.5 describes the ADRG selection data file.

ADRG CDROM areas of coverage are colored either red or green. Red indicates that the ADRG CDROM has either not yet been processed into a CAC or that the ADRG CDROM is an update of a previous edition. Green indicates that the CDROM has been processed into a CAC.

Once the desired area of coverage to be processed for a CAC is known, all the ADRG CDROMs that comprise this area can be selected for processing. A single click with mouse button1 is used to select an ADRG CDROM for processing. The area of coverage turns blue when it is selected (Fig. B26).

Information about selected ADRG CDROMs (i.e., NIMA stock number and polygon bounding coordinates) can be printed via the *Print_Screen* option. NIMA stock numbers are used to identify which ADRG CDROMs are to be used for CAC processing. This information can also be read by DISPLAY_SEGMENTS_X to display the actual coverage (for more details, see App. B, DISPLAY_SEGMENTS_X, p. 80, this report).

If PLOT_ODI is used for selecting ADRG CDROMs for processing, then the processing thread must already exist and be assigned *prior to* program invocation. The assignment or nonassignment

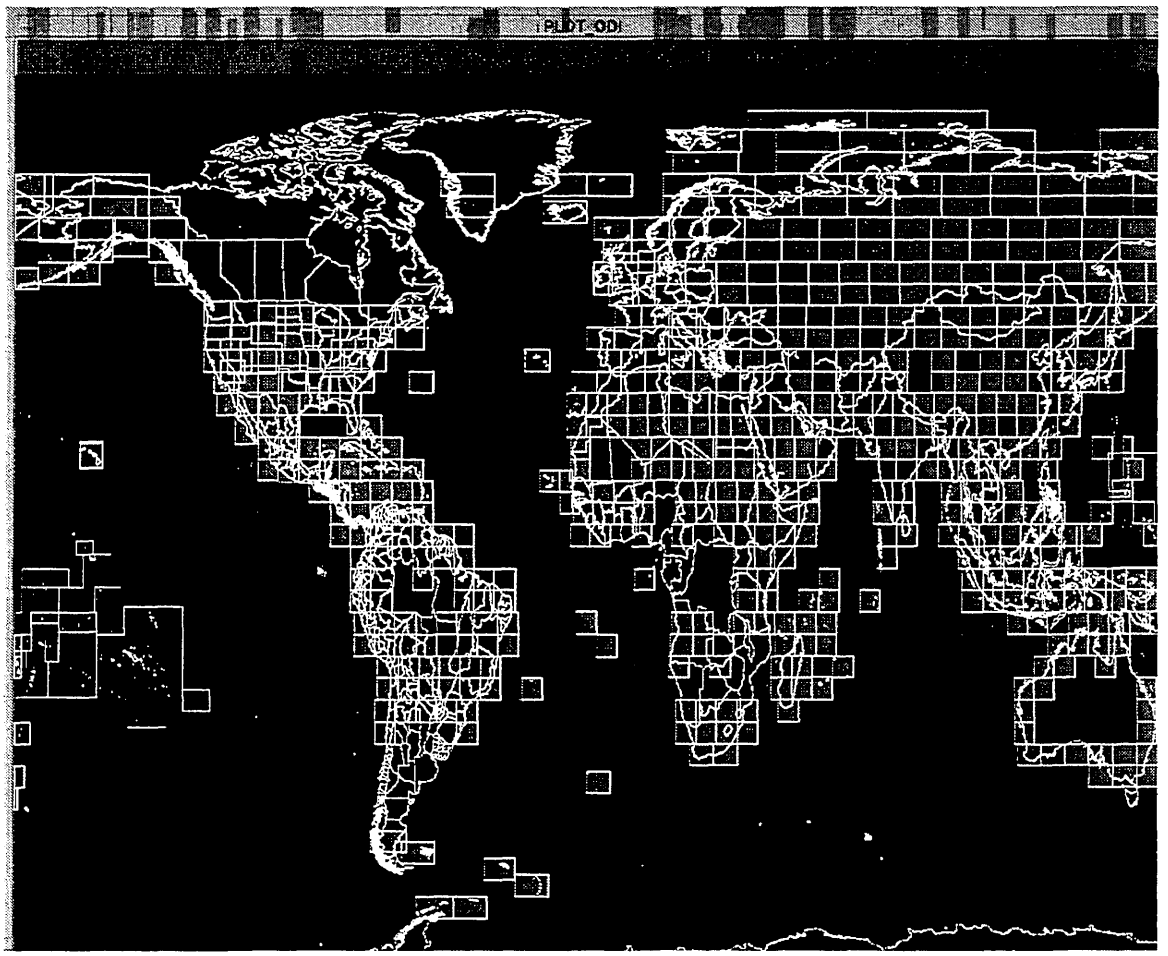


Fig. B26 — Display of ADRG coverages with colored indicators: red indicates the ADRG CDROM has not yet been processed into a CAC, green indicates the CDROM has been processed into a CAC, and blue indicates the CDROM has been selected for processing

of a processing thread determines *where* the ADRG selections data file (which contains ADRG CDROM selection information) is stored (see Sec. B6.5.2).

For future reference, a printout of the display area using the main menu bar *PRINT_SCREEN* button must be made. This printout is usually included in the processing logbook.

## B6.3 Menu Bar Options

This section covers the three menu options that are selected by a single click of mouse button1 (Fig. B27): the *FILE* option (Sec. B6.3.1), the *SELECTIONS BOX* option (Sec. B6.3.2), and the *PRINT_SCREEN* option (Sec. B6.3.5).

### *B6.3.1 File Option*

When selected, a cascade of three buttons appears (Fig. B28): the *Scale* button, the *Print ADRG Selections* button, and the *QUIT* button. Their functions are described in the following sections.

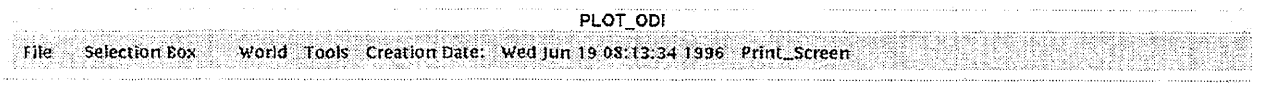



Fig. B27 — Main menu options

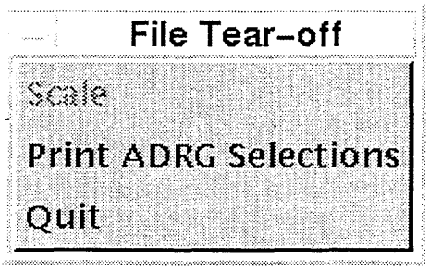



Fig. B28 — The file option cascade of buttons

### B6.3.1.1 Scale Button

Selection of this button invokes the *SCALE SELECTION* window (Fig. B29). Select a scale by highlighting the desired scale and double clicking mouse button1 (or with a single click and then selecting the "APPLY" button). PLOT_ODI now reads and displays all the ADRG CDROMs for the selected scale. To display the data, select the *DISPLAY ADRGs* button from the World selection list (Sec. B6.3.3.6).

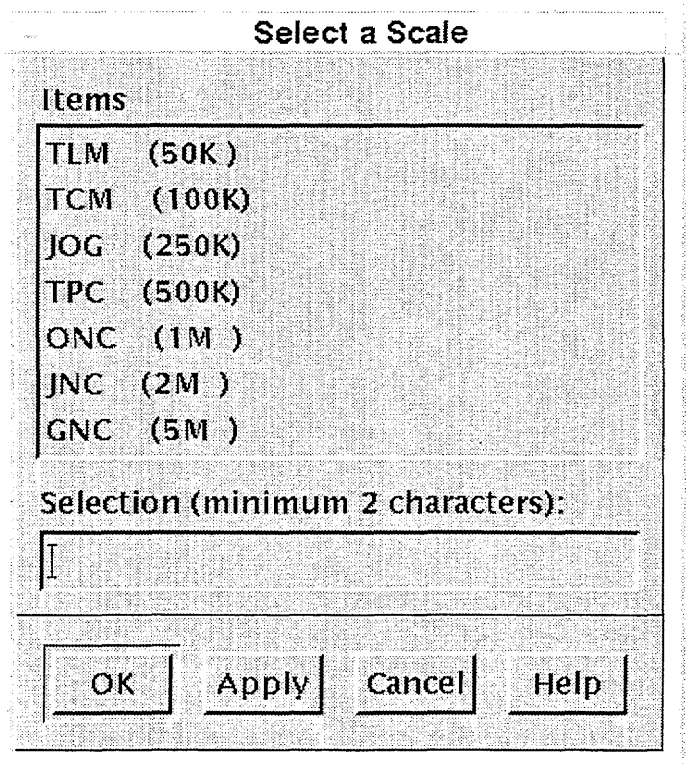



Fig. B29 — Chart scale selection window

#### B6.3.1.2 Print Button

This button is only available for use when ADRG CDROMs have already been selected for processing. This button writes the ADRG CDROM selections in the selection file (Sec. B6.5) and then prints it.

#### B6.3.1.3 Quit Button

Select this button to exit PLOT_ODI.

### *B6.3.2 Selection Box Option*

This option displays a scrollable list of all currently selected ADRG CDROMs (Fig. B30) by their NIMA stock numbers. This list will only appear if ADRG CDROMs are selected.

### *B6.3.3 World Option*

This option provides a cascade of six buttons (Fig. B31) whose functions are described in the following sections.

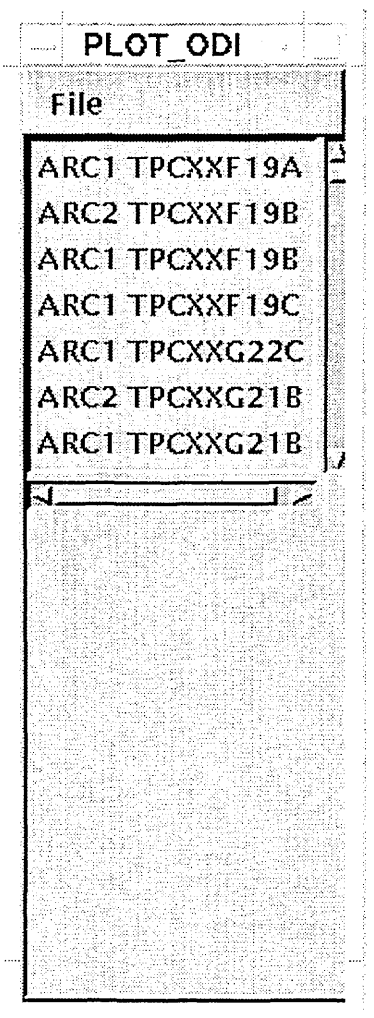



Fig. B30 — ADRG selection box

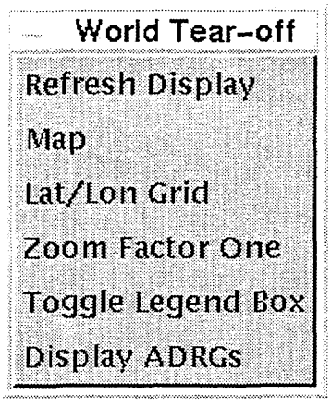



Fig. B31 — The World option cascade of buttons

#### B6.3.3.1 Refresh Display Button

When ADRG CDROMs are selected, their color changes from red or green to dark green. When deselected, their color changes from dark green back to their original color. These color changes overlay WVS and the political boundaries. To refresh WVS and the political boundaries, select the main menu *FILE* option followed by the *REFRESH DISPLAY* button. WVS and the political boundaries will then be redisplayed.

#### B6.3.3.2 Map Button

This button toggles the WVS overlay. The first selection of this button displays WVS, a second selection removes it.

#### B6.3.3.3 Lat/Lon Grid Button

This button toggles the latitude/longitude grid display. Each time it is selected, the option to change the grid spacing is provided.

#### B6.3.3.4 Zoom Factor One Button

Select this button to return to the original display size.

#### B6.3.3.5 Toggle Legend Box Button

Select this button to turn on/off a legend box in the lower right corner of the screen.

#### B6.3.3.6 DISPLAY ADRGs Button

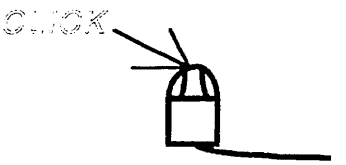
Select this button to toggle the display of selected ADRG CDROMs.

### *B6.3.4 Tools Option*

Selection of the *TOOLS* option provides a cascade of two buttons: *Zoom In* and *Zoom Out.* Their functions are described in the following sections.

### B6.3.4.1 Zoom In Button

For some chart scales (e.g., TCM and TLM), the areas of coverage are so small that they appear initially almost as dots on the display. Hence, ADRG CDROM selection/deselection is difficult to perform. The *Zoom In* function is used to address this problem (Fig. B32).




A single mouse button2 click zooms in the area that lies around the cursor's current position on the screen. The zoom in function will place the cursor's position, at the time of the zoom, in the center of the display. For example, if the cursor is placed in a general area of the Middle East (Fig. B33), that point will be centered in the display. Subsequent clicks of mouse button2 zoom in the area again (i.e., each consecutive click results in another zoom in). Continue to zoom until the coverage areas are a usable size.

### B6.3.4.2 Zoom Out Button

Zoom out operation is performed by single clicking mouse button3. The *Zoom Out* button performs the opposite function as the *Zoom In* button. For example, if the Zoom In function was

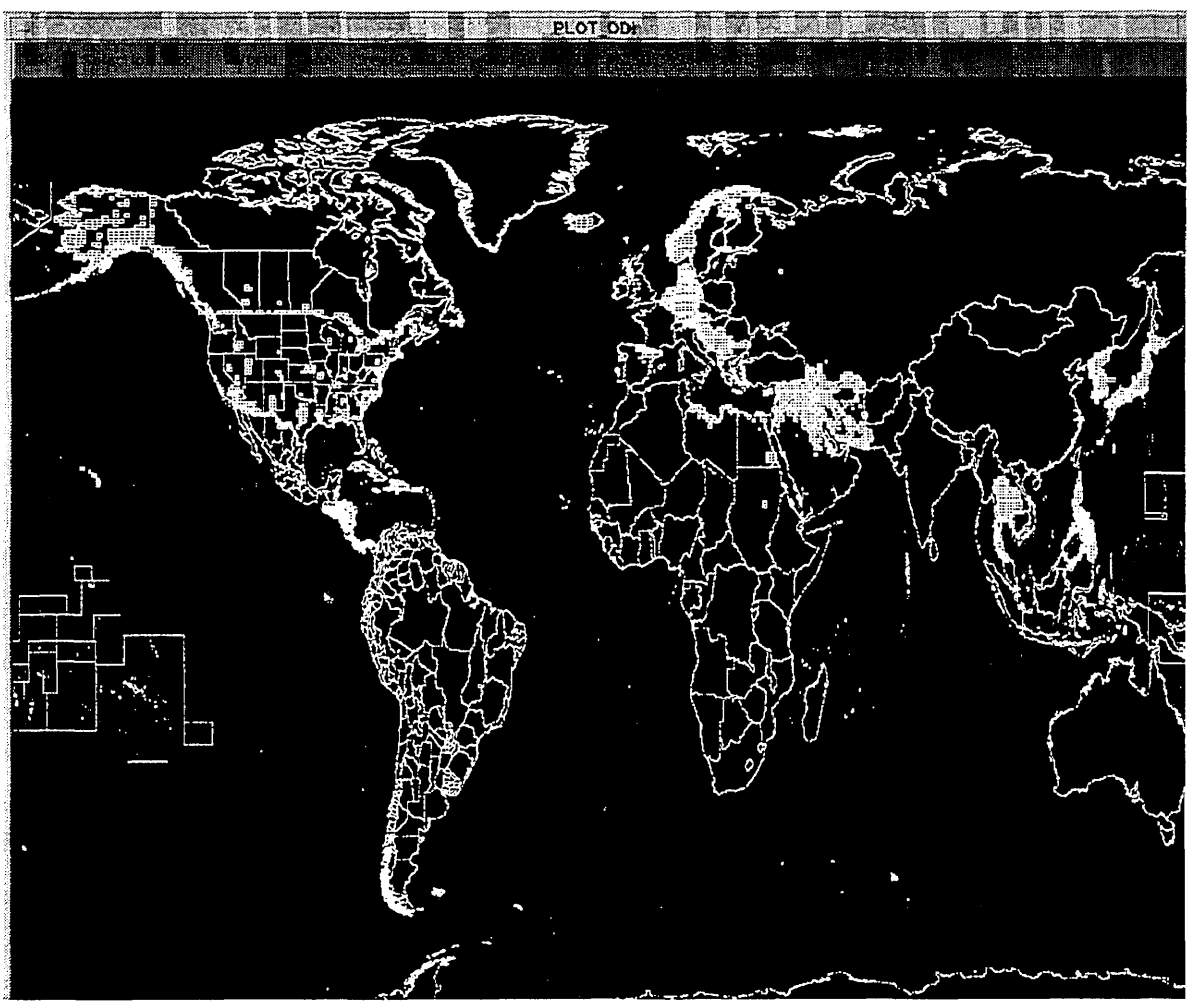



Fig. B32 — Example of TML chart coverages

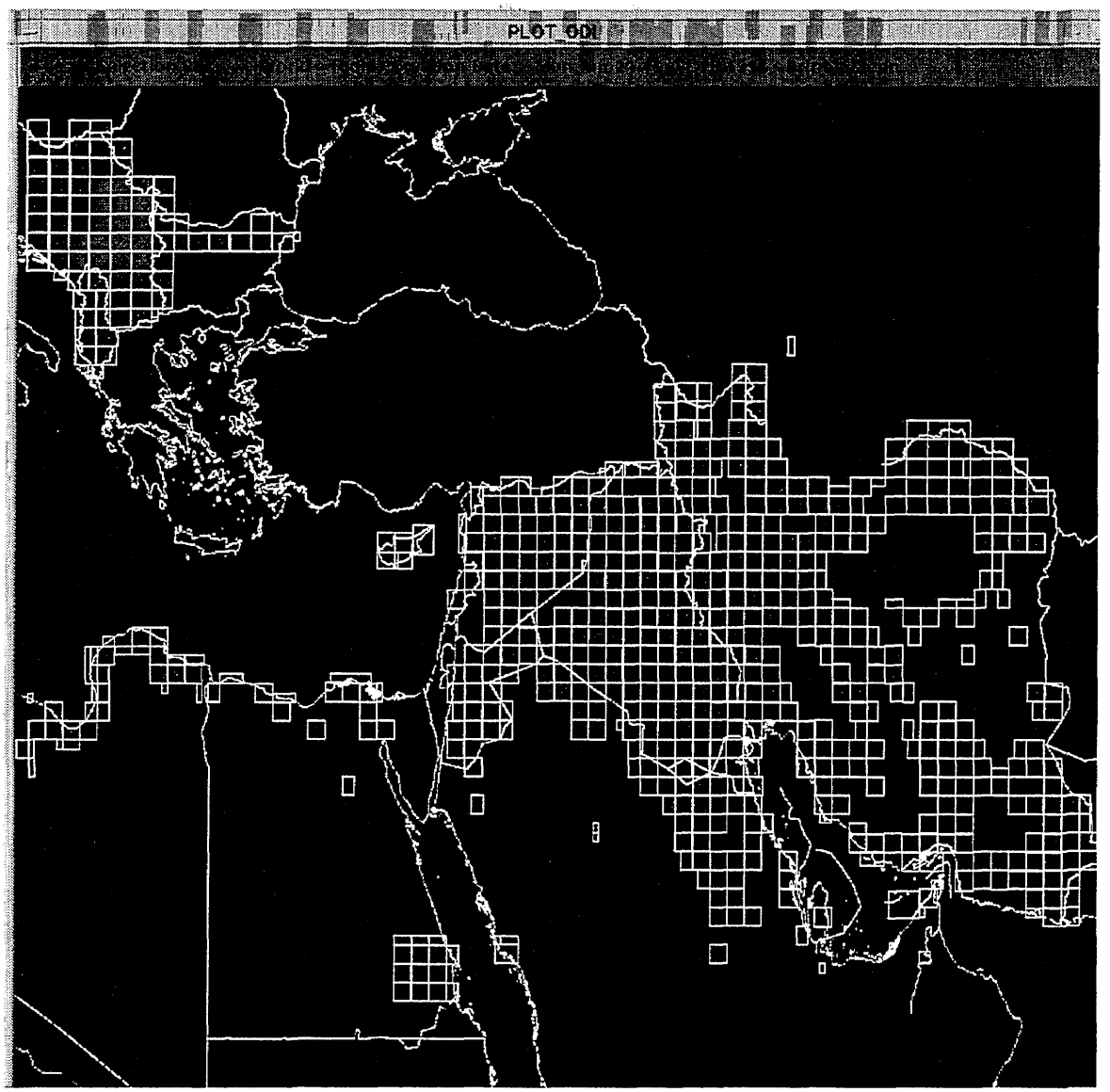



Fig. B33 — Zoomed in area of TLM chart

performed four times, then one Zoom Out reduces the image to the equivalent of a three times Zoom In. A Zoom Out operation can only be performed as many times as the Zoom In operation (i.e., once back to the original size, Zoom Out has no further effect).

*B6.3.5 Print_Screen Option*

Selection of this option prints a plot file of the display area on the TEKTRONIX printer.

## B6.4 Mouse Button Functions

This section describes the three mouse button functions: Sec. B6.4.1 covers the selection/ deselection of ADRG CDROMs and Sec. B6.4.2 covers the coordinates functions.

*B6.4.1 Mouse Button1: ADRG CDROM Selection*

Mouse button1 is used for selecting and deselecting ADRG CDROMs.

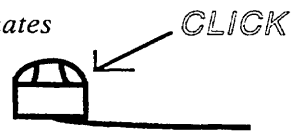
- To select a CDROM:

  Place the cursor inside the coverage of an ADRG CDROM (colored either red or green) and single click mouse button1. The colored area changes to a dark green, indicating that the ADRG CDROM is selected.

- To deselect a CDROM:

  With cursor inside a dark green area, double click mouse button1. The dark green changes to either red or green, indicating that the ADRG CDROM is deselected.

*B6.4.2 Mouse Button3: Coordinates*




Mouse button3 (i.e., the right-most button on a mouse) is used to obtain information about a segment on the display. Move the cursor to a segment on the display and single click mouse button3. A window appears on the upper left of the display containing the latitude/longitude coordinates of the pixel that the cursor lies upon, the row and column values for the segment, the latitude/longitude coordinates of the lower-left (southwest) corner of the segment, and the segment key name. Note that no information is available for segments outside of the selected TS zone.

## B6.5 THE ADRG SELECTION FILE

This section describes the ADRG selection data file: Sec. B6.5.1 specifies information that is contained in the file and Sec. B6.5.2 describes the naming convention and location of the file (both depend on whether or not a processing thread was assigned prior to invocation).

### *B6.5.1 Selection File Contents*

The ADRG selection data file contains the following information:

| Field | Description |
|---|---|
| 1 | Entry number–the sequence in which the ADRG CDROM was selected (i.e., 0,1,2...) |
| 2 | NIMA stock number |
| 3 | Edition number |
| 4 | Northern latitude (NL) |
| 5 | Southern latitude (SL) |
| 6 | Western-most longitude (WL) |
| 7 | Eastern-most longitude (EL) |

Figure B34 shows a sample listing of the information the file would contain if all 26 entries for Global Navigation and Planning Charts (GNC) were selected.

*B6.5.2 ADRG Selection File Name and Location*

The data file name and location is dependent on whether a processing thread has been assigned prior to executing PLOT_ODI. Section B6.5.2.1 describes the case where a thread was assigned and Sec. B6.5.2.2 describes the case where a thread was not assigned.

### B6.5.2.1 Processing Thread Assigned

When a processing thread was assigned prior to executing PLOT_ODI, the file name will be MDFF_SCRATCH:[000000] XXX_ADRG_SELECTIONS.DAT (where XXX is the processing thread name). Its contents can be listed by typing the following command:

*Type* **MDFF_SCRATCH:[000000]XXX_ADRG_SELECTIONS.DAT.**

The output will appear as shown in Fig. B34 (without the top line that contains headings).

### B6.5.2.2 No Processing Thread Assigned

When a processing thread was assigned prior to running PLOT_ODI, the file will be named ADRG_SELECTIONS.DAT and it will be written to the user's home directory via the logical name "SYS$LOGIN." Type the following command to list the file:

*Type* **SYS$LOGIN:ADRG_SELECTIONS.DAT**

and the output will appear as shown in Fig. B34 (without the top line that contains headings).

| ENTRY | STOCK NUMBER | EDITION | NL | SL | WL | EL |
|---|---|---|---|---|---|---|
| 0 | ARC4 GNCXX01 | 1 | 68.00 | 90.00 | -180.00 | 180.00 |
| 1 | ARC4 GNCXX06 | 1 | 40.00 | 68.00 | 145.00 | -130.00 |
| 2 | ARC4 GNCXX05 | 1 | 40.00 | 68.00 | 70.00 | 154.00 |
| 3 | ARC4 GNCXX04 | 1 | 40.00 | 68.00 | -10.00 | 70.00 |
| 4 | ARC4 GNCXX03 | 1 | 40.00 | 68.00 | -70.00 | -10.00 |
| 5 | ARC4 GNCXX02 | 1 | 24.00 | 68.00 | -130.00 | -65.00 |
| 6 | ARC4 GNCXX08 | 1 | 0.00 | 40.00 | -160.00 | -100.00 |
| 7 | ARC4 GNCXX07 | 1 | 0.00 | 40.00 | -179.60 | -156.00 |
| 8 | ARC4 GNCXX09 | 1 | 0.00 | 40.00 | -100.00 | -50.00 |
| 9 | ARC4 GNCXX10 | 1 | 0.00 | 40.00 | -50.00 | -15.00 |
| 10 | ARC4 GNCXX11 | 1 | 0.00 | 40.00 | -15.00 | 40.00 |
| 11 | ARC4 GNCXX12 | 1 | 0.00 | 40.00 | 40.00 | 90.00 |
| 12 | ARC4 GNCXX13 | 1 | 0.00 | 40.00 | 90.00 | 145.00 |
| 13 | ARC4 GNCXX20 | 1 | -40.00 | 2.00 | 165.00 | -155.00 |
| 14 | ARC4 GNCXX14 | 1 | -40.00 | 0.00 | 110.00 | 165.00 |
| 15 | ARC4 GNCXX15 | 1 | -40.00 | 0.00 | 68.00 | 110.00 |
| 16 | ARC4 GNCXX16 | 1 | -40.00 | 0.00 | 10.00 | 68.00 |
| 17 | ARC4 GNCXX17 | 1 | -40.00 | 0.00 | -40.00 | 10.00 |
| 18 | ARC4 GNCXX18 | 1 | -40.00 | 0.00 | -95.00 | -40.00 |
| 19 | ARC4 GNCXX19 | 1 | -40.00 | 0.00 | -155.00 | -95.00 |
| 20 | ARC4 GNCXX25 | 1 | -75.00 | -40.00 | -178.00 | -125.00 |
| 21 | ARC4 GNCXX24 | 1 | -75.00 | -40.00 | -125.00 | -50.00 |
| 22 | ARC4 GNCXX26 | 1 | -90.00 | -75.00 | -180.00 | 180.00 |
| 23 | ARC4 GNCXX23 | 1 | -75.00 | -40.00 | -50.00 | 20.00 |
| 24 | ARC4 GNCXX22 | 1 | -75.00 | -40.00 | 20.00 | 95.00 |
| 25 | ARC4 GNCXX21 | 1 | -75.00 | -40.00 | 95.00 | -178.00 |

Fig. B34 — ADRG SELECTIONS.DAT file listing

# SQCODE

## B7.0 SQCODE

Use the Sqcode utility to list current generic batch and codebook compression queue assignments:

*Sqcode*, where there are no parameters.

Figure B35 provides a sample Sqcode listing that includes the generic batch queues for processing threads A2A (A2A_CODEBOOKS) and A2C (A2C_CODEBOOKS). Associated codebook compression queues for A2A (CODEBOOK1 and CODEBOOK2) are also shown. Job file names in compression queues use the following convention:

xxx_Immmmmssss,

where

xxx = a processing thread name,
I = a processing phase indicator. "C" indicates PASS1 compression, "P" indicates PASS3 compression,
mmmmm = a five-digit MDFF sequence number, and
ssss = a four-digit codebook status file entry.

Generic batch queue A2A_CODEBOOKS

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 458 | A2A_C012340013 | CAC | Pending |
| 459 | A2A_C012340014 | CAC | Pending |
| 460 | A2A_C012340015 | CAC | Pending |
| 461 | A2A_C012340016 | CAC | Pending |
| 462 | A2A_C012340017 | CAC | Pending |
| 463 | A2A_C012340018 | CAC | Pending |
| 464 | A2A_C012340019 | CAC | Pending |
| 465 | A2A_C012340020 | CAC | Pending |

Generic batch queue A2C_CODEBOOKS

Batch queue CODEBOOK1, busy, on ZOMBIE::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 446 | A2A_C012340001 | CAC | Executing |
| 447 | A2A_C012340002 | CAC | Executing |
| 448 | A2A_C012340003 | CAC | Executing |
| 454 | A2A_C012340009 | CAC | Executing |
| 455 | A2A_C012340010 | CAC | Executing |
| 456 | A2A_C012340011 | CAC | Executing |

Batch queue CODEBOOK2, busy, on ZOMBIE::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 449 | A2A_C012340004 | CAC | Executing |
| 450 | A2A_C012340005 | CAC | Executing |
| 451 | A2A_C012340006 | CAC | Executing |
| 452 | A2A_C012340007 | CAC | Executing |
| 453 | A2A_C012340008 | CAC | Executing |
| 457 | A2A_C012340012 | CAC | Executing |

Fig. B35 — Sample Sqcode listing. Note that processing thread names are incorporated into job names.

# SQMDFF

## B8.0 SQMDFF

MDFF_BATCH queue activity is monitored by the Sqmdff utility:

*Sqmdff*, where there are no parameters.

Figure B36 provides an example of output.

Batch queue MDFF_BATCH, available, on ZOMBIE::

| Entry | Jobname | Username | Status |
|---|---|---|---|
| 897 | MDFF_CONTROL | SYSTEM | Executing |
| 351 | A2A_PROCESSCHRT | CAC | Executing |
| 534 | A2C_PROCESSCHRT | CAC | Executing |

Fig. B36 — Sample Sqmdff output listing

# THREAD

## B9.0 THREAD

Processing threads are created during the CAC Processing INITIALIZATION phase (Sec. 4.0) and use a standard three-character code:

- The first character distinguishes the data set type; for CAC an "A" *must be used* (for aeronautical chart).
- The second character identifies the scale of the data set [(0)-1:50K, (1)-1:100K, (2)-1:250K, (3)-1:500K, (4)-1:1M, (5)-1:2M, and (6)-1:4M]. *The only valid* values to use are between 0 and 6.
- The third character is a sequence identifier [(A)- first, (B)-second, etc.]. This is used to delineate simultaneous CAC ODI builds when both charts are at the same scale.

For example, the processing thread *A4A* denotes a CAC ODI build for an (*A*) aeronautical chart at the (*4*) ONC (1:1M) scale with sequence identifier (*A*).

Since multiple CACs can be processed concurrently, a processing thread must be defined to access chart data that is associated with a particular CAC ODI build. The *current processing thread* is the thread that was *most recently* defined. Use the Thread utility to define an existing processing thread:

*thread* **xxx**, where xxx = processing thread name.

# CAC_QC CONTENTS

B10.0 INTRODUCTION ..... 114
B10.1 Invocation ..... 114
B10.2 Processed Data Review ..... 114
B10.2.1 PASS1 Processing ..... 114
B10.2.1.1 Chart Data Review ..... 114
B10.2.1.2 Legend Data Review ..... 117
B10.2.1.3 Zoom Feature ..... 117
B10.2.1.4 Tessellate Feature ..... 117
B10.2.1.5 Planet Mode ..... 118
B10.2.1.6 Other Features ..... 119
B10.2.1.7 Quitting CAC_QC ..... 119
B10.2.2 PASS3 Review ..... 120
B10.2.2.1 PASS3A Review ..... 120
B10.2.2.2 PASS3B Review ..... 121
B10.2.3 Full Review ..... 121
B10.3 CAC CDROM Review ..... 122
B10.3.1 CAC Data Review ..... 122
B10.3.2 CAC Legend Data ..... 123
B10.4 Color Palette Review ..... 123
B10.4.1 Standard Color Palettes ..... 123
B10.4.2 Algebraic Color Palette ..... 123
B10.4.3 Color Palette of a Palette Creation Thread ..... 124
B10.4.4 Reviewing Downsampled Segments ..... 124

# CAC_QC

## B10.0 INTRODUCTION

CAC_QC is the program used for quality control including review of data being processed in a processing thread, review of data from CAC CDROM, and viewing CAC color palettes.

### B10.1 Invocation

Use of the following command to invoke CAC_QC:

CAC_QC.

An optional qualifier is provided for reading from a CDROM device:

CAC_QC **cdreader#**, where **cdreader#** = an optional qualifier that specifies the CDROM reader containing the CAC CDROM.

### B10.2 Processed Data Review

*B10.2.1 PASS1 Processing*

The main function of CAC_QC is to review processed ADRG CDROMs. As an ADRG CDROM is selected for processing, it is entered into the CHART_STATUS.DAT file with an initial status of NO_OP. CAC_QC cannot review these CDROMs until they are marked as "CD_COMPLETE." During PASS1, only core segments of each ADRG CDROM are compressed and viewable. Legend data are processed and reviewed after the ADRG CDROM has completed processing.

Section B10.2.1.1 describes review of chart data for the PASS1 and Sec. B10.2.1.2 describes legend data processing. Section B10.2.1.3 describes the zoom feature and Sec. B10.2.1.4 explains the tessellation feature. Viewing data at random (i.e., not dependent on ADRG CDROMs) is performed with the "PLANET" function described in Sec. B10.2.1.5. Additional features are discussed in Sec. B10.2.1.6 and program termination is described in Sec. B10.2.1.7.

#### B10.2.1.1 Chart Data Review

Several windows appear after invocation and the main review window (Fig. B37) is the largest of these. Note that only some of the options are initially available for selection since some of the functions provided by these options require more information or data than is currently available (e.g., the PASS3B button cannot be selected until PASS3A has completed).

To review an ADRG CDROM that has completed processing, select the *CD_OPTION* option by a single click of the left-most mouse button (hereafter referred to as button1). The following cascade buttons appear: *RESET CD, SELECT CD, REVIEW, TESSELATE,* and *QUIT.*

Fig. B37—Main menu options

Using a single button1 click, choose the *SELECT CD* option. A text window containing names of the processed ADRG CDROMs appears in the middle of the screen (Fig. B38). If the ADRG CDROM to be reviewed is not visible, use the scroll bar and scroll through the text until the desired CDROM appears. Select this CDROM by double clicking button1. The selection window disappears, leaving the main review window still blank.

To begin reviewing chart data, place the cursor in the review window and click button1. Alternatively, use button1 to select the *CD_OPTIONS* option and then select the "REVIEW" option (which is now selectable).

The cursor changes to a "watch" symbol while data are read and decompressed for display. Afterwards, data appear in the review window (Fig. B39) and the cursor reverts to an "arrow" symbol. After the data are reviewed and deemed satisfactory, click button1 to review the next set of segments. Data are advanced from left to right, with an overlap of one column from the previous screen. When the end of data is encountered (on the right), the columns appear black. Review then restarts at the beginning column.

Continue reviewing data by repeatedly placing the cursor in the review window and clicking button1. Progression of the review can be monitored by selecting the *SEGMENT* option on the main menu bar: a window will appear on the upper-left portion of the monitor that displays a wire frame of the segments for the ADRG CDROM being reviewed (Fig. B40). As a segment is decompressed

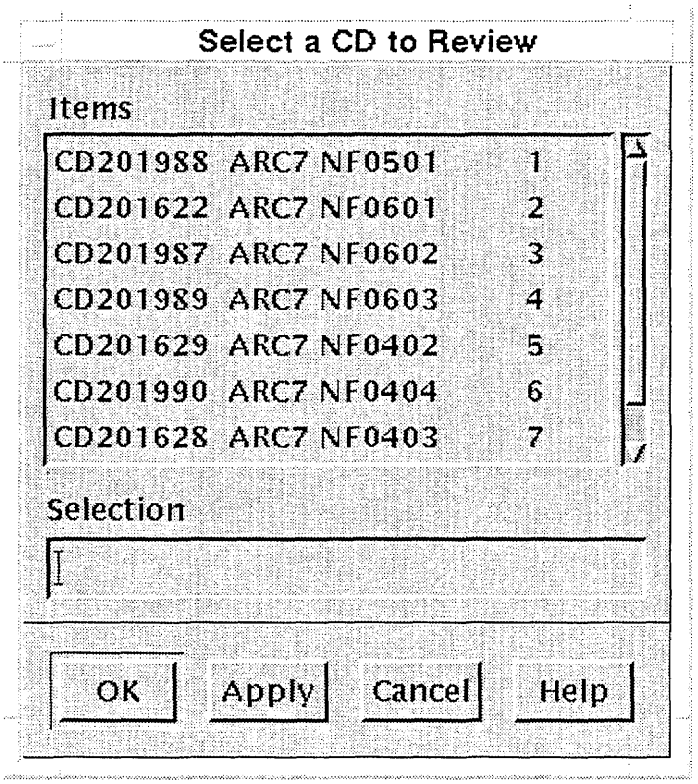



Fig. B38 — Text window containing the names of processed ADRG CDROMs

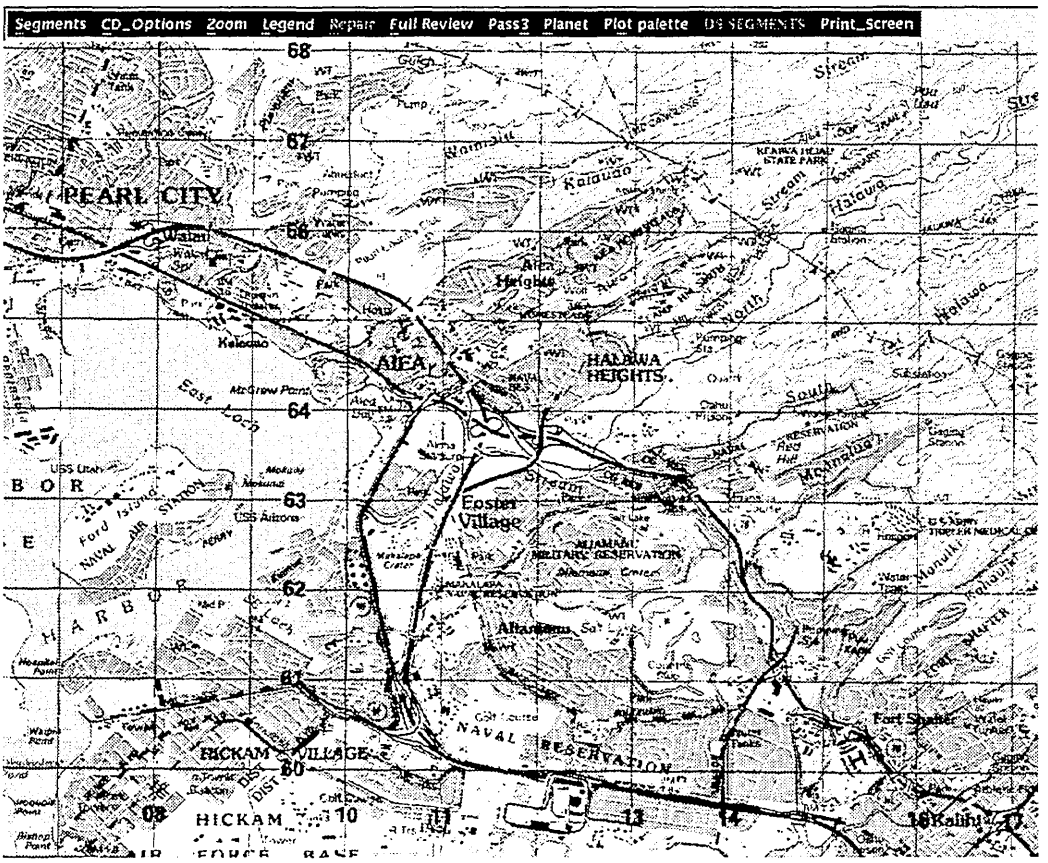



Fig. B39 — CAC data display

and displayed, its corresponding square will turn white within the wire frame. Segments that have already been reviewed are yellow. This window can be toggled on and off with the *SEGMENT* option.

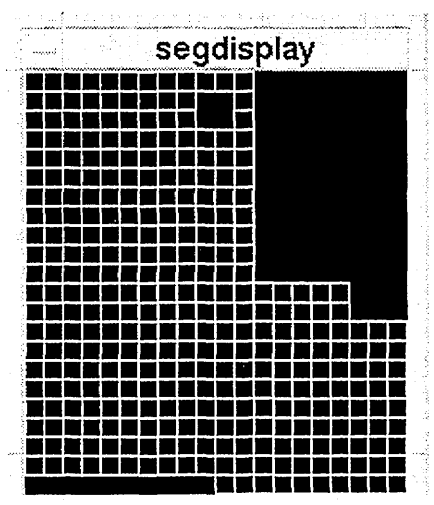



Fig. B40 — Reviewed segments wire frame

After all segments have been reviewed, an "End of Zone" window appears with instructions to reset and reselect the ADRG CDROM to review data in the next zone (Fig. B41). Zones must be reviewed separately since each zone has its own color palette and chart data may overlap zones.

To review another zone, click on the "OK" option in the "End Zone" window. Next, click on the CD_OPTIONS button in the main menu bar and select the *RESET* option to reinitialize CAC_QC. Now, reselect the *CD_OPTIONS* option and then the *SELECT_CD* option. The CDROM selection window reappears with the just-reviewed CDROM highlighted. Reselect this CDROM by double clicking button1. If there are data in other zones, review this data as previously described. Otherwise, review of data for this CDROM is complete. An "End of Data" message box will appear. Use button1 to select the *OK* option and dismiss the message box. The legend data for this CDROM must also be reviewed (see the following section).

To review another ADRG CDROM, re-initialize CAC_QC and repeat the steps described earlier in this section.

### B10.2.1.2 Legend Data Review

Legend data for a given CDROM can be reviewed by selecting the *LEGEND* option from the main menu bar. Once selected, the first legend file will be displayed (Fig. B42). To review the next legend file, either select the *LEGEND* option again or click button1 with the cursor placed in the review window. After all the legend files have been reviewed, an "End of Legend" message box will appear. Select the *OK* option to dismiss the message box.

Review of an ADRG CDROM is complete after all the chart and legend data have been reviewed.

Note that the menu bar obscures some of the legend and chart data that is being displayed. The menu bar can be removed by clicking the third right-most mouse button (called button3) to reveal the obscured data in the display. Reclick button3 to return the menu bar.

### B10.2.1.3 Zoom Feature

This feature allows chart data to be zoomed out (i.e., the data appear to be "demagnified"). Once selected, a cascade of optional zoom factors appears. To choose a zoom factor, highlight a number and click button1. To display the data, place the cursor in the review window and click button1.

A zoom factor should be selected *before* starting chart data review since some data may be inadvertently missed due to the row and column minimum/maximum value calculations that occur when a zoom factor is selected. The recommended course of action is to first select a CDROM to review, then select a zoom factor and begin reviewing the data.

Zoom factor 1 is sufficient for chart data review. A zoom factor *greater than 2* is not recommended, since this feature was designed to provide quick overviews of large areas and for general browsing.

### B10.2.1.4 Tessellate Feature

By default, chart data are displayed with visible segment boundaries (i.e., tessellations) as shown in Fig. B43. The

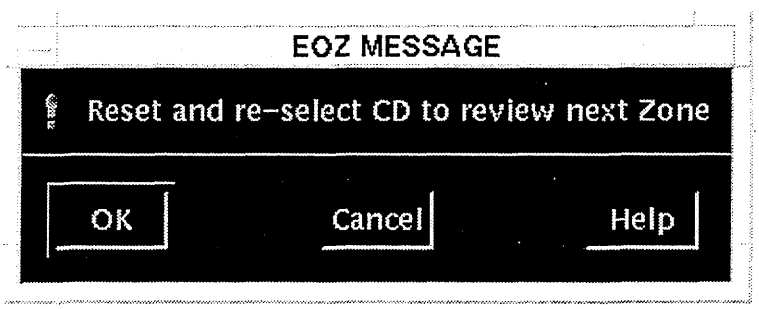



Fig. B41 — End of zone window

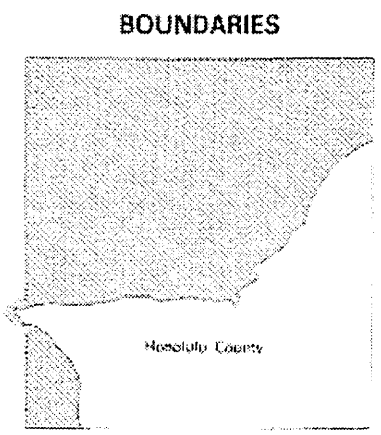



Fig. B42 — Example of a legend data file

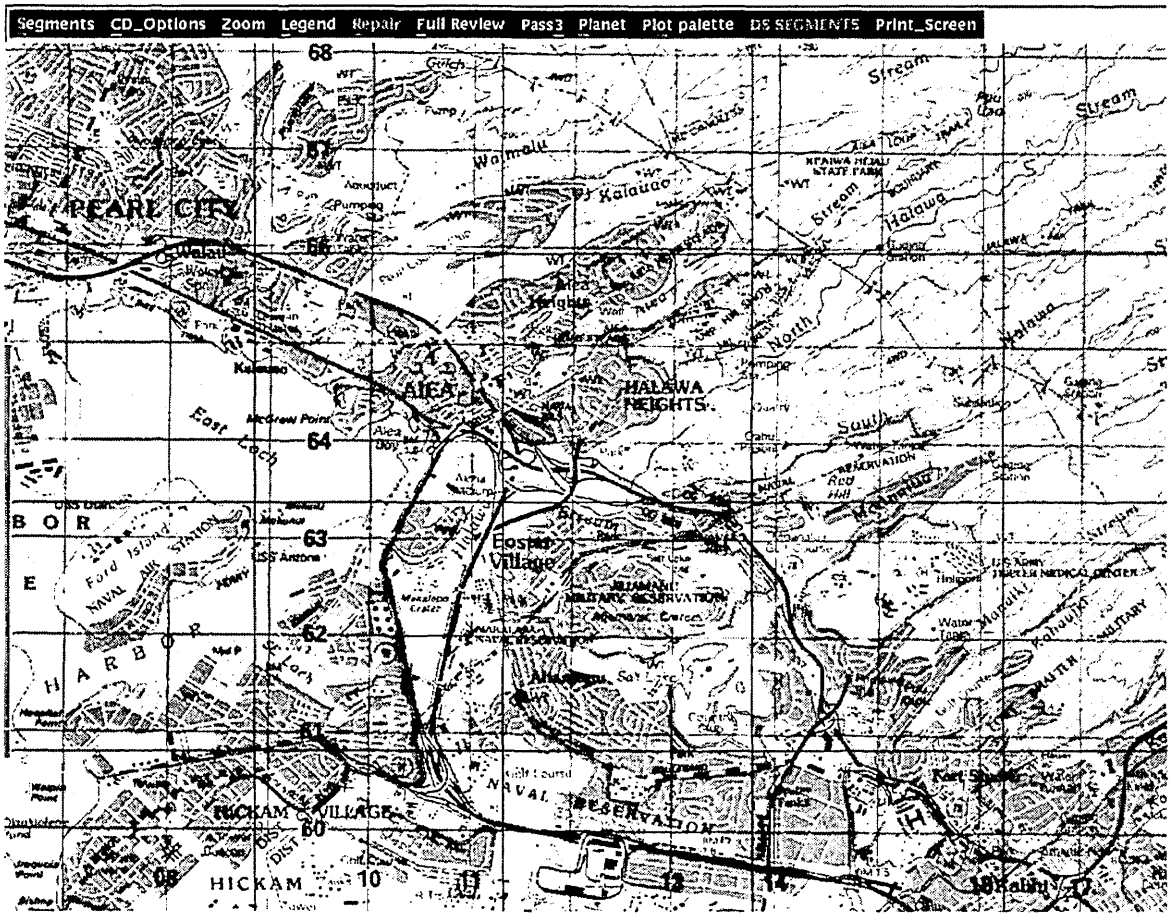



Fig. B43 — CAC data display with tessellation

display area can be set to omit these borders (i.e., untessellated, Fig. B39) by first selecting the *CD_OPTIONS* option on the main menu bar and then selecting one of the following *TESSELLATE* options:

NO turns off the tessellation
YES turns on the tessellation

### B10.2.1.5 Planet Mode

This mode is used to view data that have been processed in the current processing thread or from a CAC CDROM. For example, to view an area around Paris, France, first determine the row and column numbers for this area (e.g., via CAC_CALC). If this area of coverage is not contained within the current processing thread, it can be loaded from the appropriate CAC CDROM.

Select PLANET mode and a window will appear to obtain the row/column and zone values (Fig. B44). Highlight the text box below "Select Row" and enter a row number. Enter the column and zone numbers by highlighting the appropriate selection boxes and entering values. Now, select the *INVOKE* option and data will then be displayed.

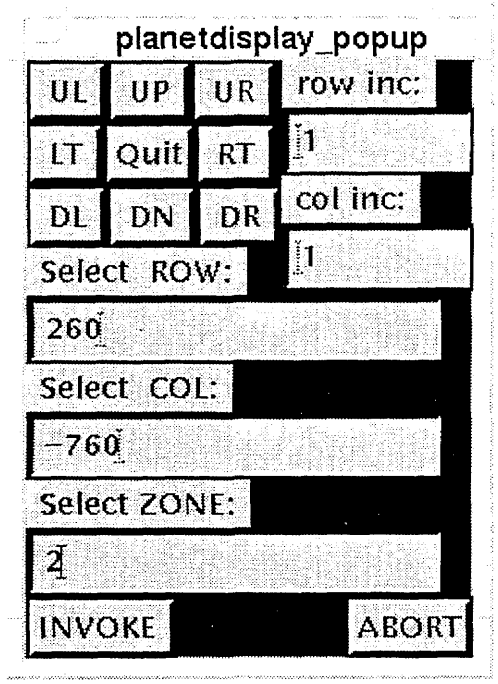
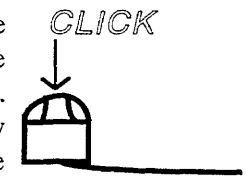



Fig. B44 — Planet mode window for obtaining row, column, and zone values




Place the cursor in the main window and click mouse button2, then select the *PLANET* option. Note that the row and column values have changed; these are the row and column values of the segment on which the cursor currently resides. Select *INVOKE* to display the new area. The data display can be scrolled by pressing one of the directional buttons in the *PLANET* window (Fig. B44). The row and column scrolling increments are both set to one.

### B10.2.1.6 Other Features

Mouse button2 is used to obtain information about segment coordinates by placing the cursor in the review window over a segment and clicking mouse button2. The following information about the segment (on which the cursor lies) is displayed in the review window: TS row and column numbers, pixel latitude and longitude coordinates, and the lower-left (southwest) corner latitude and longitude of the segment (Fig. B45).

Click button2 again to remove the information or move the cursor to another segment to obtain information about that segment. Note that in the segment review window (see Sec. B10.2.1.1) the corresponding segment in the wire frame turns blue, indicating that it has been reviewed.

The cursor can be placed on *any segment* in the wire-frame window and with button2 used to display the same coordinate information in the user's main default terminal window (icon the review window to see this information). Note that these segments' corresponding wire-frame square turns red, indicating that they have been selected.

### B10.2.1.7 Quitting CAC_QC

To quit the program, select CD_OPTION from the main menu bar and click on the QUIT option with mouse button1. All windows associated with CAC_QC will disappear as CAC_QC is terminated.

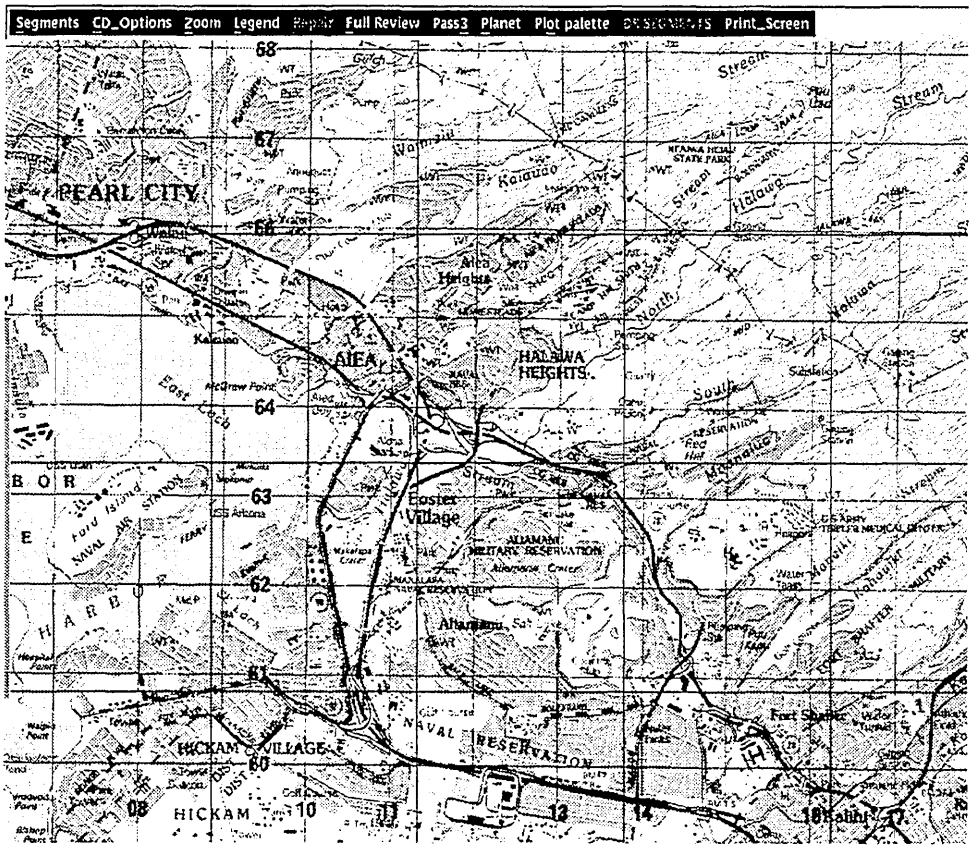



Fig. B45 — Segment information produced by mouse button2 including TS row and column numbers, pixel latitude and longitude coordinates, and the lower left (southwest) corner latitude and longitude

*B10.2.2 PASS3 Review*

PASS3 consists of two processing steps; processing of edge filled segments (PASS3A) and processing of all remaining unprocessed segments (PASS3B). In the sections that follow, Sec. B10.2.2.1 describes reviewing of PASS3A processed segments and Sec. B10.2.2.2 covers the reviewing of PASS3B segments.

**B10.2.2.1 PASS3A Review**

Select the *PASS3* option from the main menu to begin reviewing data. Available options include:

- *PASS3A*
- *PASS3B* (which is selectable only after PASS3A is complete)
- *QUIT*

Select the PASS3A option and data will appear in the review window. To determine which PASS3A segments are to be reviewed, select the *SEGMENTS* option in the menu bar to display the segment wire frame (see Sec. B10.2.1.1). Segments processed in PASS3A initially appear red (Fig. B46) and those segments that are currently being displayed appear white. Reviewed segments will appear yellow and PASS1 segments appear green.

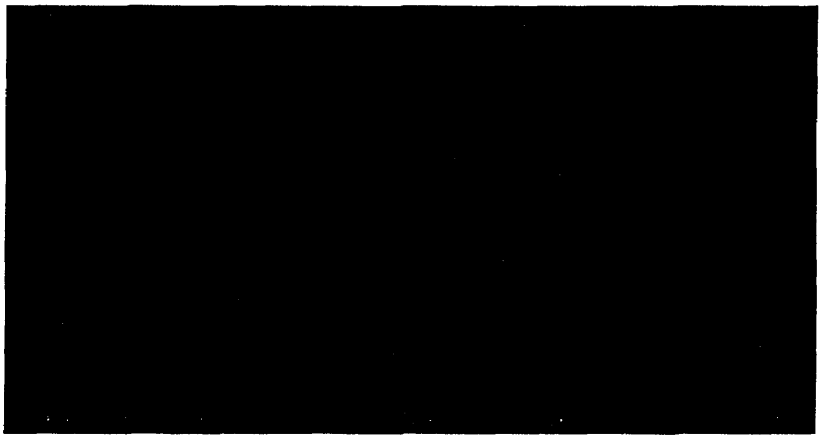


Fig. B46 — PASS3A segments wire frame showing PASS3A segments in red

An "End of Zone" message (Fig. B41) will appear when review of data in the current zone is complete. Acknowledge this message and reselect the PASS3A option to review the next zone, repeating until all zones have been reviewed. Since there is no legend data, PASS3A review is complete after all of the available data in each zone have been reviewed. The PASS3B option becomes selectable after PASS3A is complete.

#### B10.2.2.2 PASS3B Review

PASS3B review is similar to PASS3A review with the following differences:

- PASS3B segments are initially displayed blue in the segments wire-frame display and
- most PASS3B segments are partial segments.

A final full review (Sec. B10.2.3) of data must be performed after PASS3B review is complete.

### *B10.2.3 Full Review*

The purpose of a final full review is to verify that there are no missing segments and that all segments have been reviewed. The most efficient method is to view large areas of data in one zone at a time, which is accomplished by selecting the *FULL REVIEW* option from the menu bar.

Data are displayed using a zoomed-out factor of 16. Select the *SEGMENTS* option to display the segments window (Fig. B47) where segments are displayed as follows:

| *Phase* | *Color* |
|---|---|
| PASS1 | Green |
| PASS3A | Red |
| PASS3B | Blue |

Segments that are currently being displayed appear white in the segments wire-frame window. Segments that have already been reviewed during this execution are yellow. Note that since large amounts of segments are being decompressed and stored in memory, it may take 5 to 20 min to display a single window of data. However, it should only take a few displays to cover an entire zone of processed data.

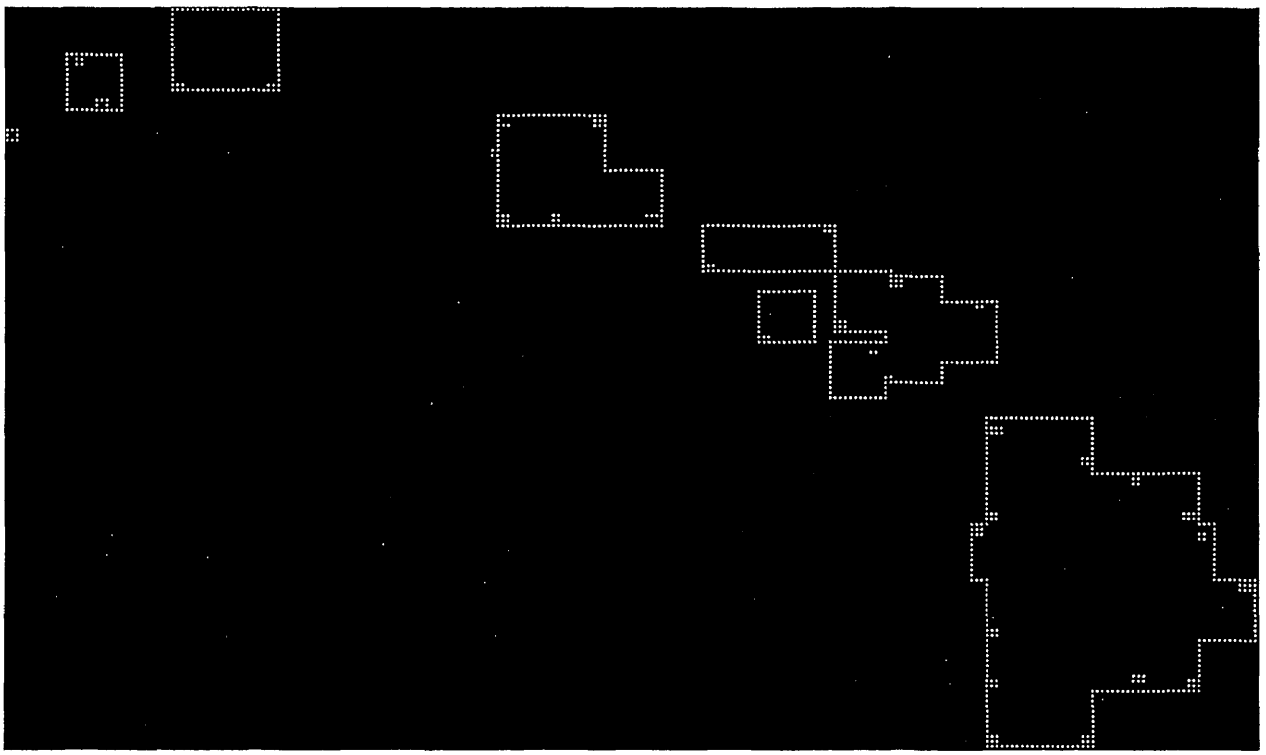


Fig. B47 — Full review segments wire frame (showing PASS3B segments)

After all the data have been reviewed and determined acceptable, the review process is complete and processing can now be finalized.

## B10.3 CAC CDROM REVIEW

CAC_QC is used to view chart and legend data on a CAC CDROM:

- Place a CAC CDROM on a CDROM reader and mount the device.
- Verify that no processing thread is assigned.
- Invoke CAC_QC:

***CAC_QC*** **device,** where **device** = CDROM reader containing the CAC CDROM.

Refer to Sec. B10.2 for information about reviewing chart and legend data.

### *B10.3.1 CAC Data Review*

The *PLANET* option must be selected for CAC review since there are no data files (e.g., DS_SEGMENTS.DAT for PASS1 and SEGMENTS.DAT for PASS3). Required inputs include TS zone, row, and column numbers for the desired area of coverage.

When the "Planet" window appears, enter the row, column, and zone values for segments to be viewed (Fig. B45). Segments from a CAC CDROM are read, decompressed, and displayed in the review window. Use the review procedure described in Sec. B10.2.1.5. If review of a large area is desired, select a "ZOOM" factor and continue reviewing.

*B10.3.2 CAC Legend Data*

To review legend data from a CAC CDROM, select the *CD OPTION* in the menu bar and then click on the *Select CD* button. Legend data review is described in Sec. B10.2.1.2.

## B10.4 Color Palette Review

CAC_QC can be used to review existing (standard) CAC color palettes, the algebraic color palette, or a color palette from a palette creation thread. For a palette creation thread, downsampled segments that will be used to generate a color palette can be reviewed before processing with the algebraic palette and then again with the newly created color palette.

*B10.4.1 Standard Color Palettes*

To view the standard CAC palettes, select the *PLOT_PALETTE* option in the main menu then click on the *OTHER PALETTE* option. A popup window appears (Fig. B48); enter the chart scale (0–5) and zone (0–4) of the palette to view and then select the *INVOKE* option. The palette (containing up to 240 colors) will be displayed. Note: starting at the upper-left corner, the first color entry in a color palette is BLACK.

The colors' intensities increase across to the right and then wrap around to the beginning of the next row down (Fig. B49). The right-most color on the last row is the last entry in color palette. For most of the color palettes, this color will be white (with the exception of a few earlier palettes). Some older color palettes may not have all 240 entries filled since the original palette generation code did not fill all available entries. However, all slots are filled in newer color palettes.

*B10.4.2 Algebraic Color Palette*

The algebraic color palette is used to review downsampled segments that have been selected for creating a new color palette. To view the algebraic color palette, select the *PLOT PALETTE* option in the main menu and then click on the *ALGEBRAIC* option. The algebraic palette is displayed as described in Sec. B10.4.1.

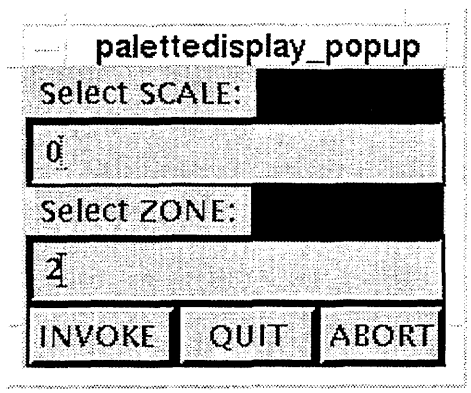



Fig. B48 — Palette display window

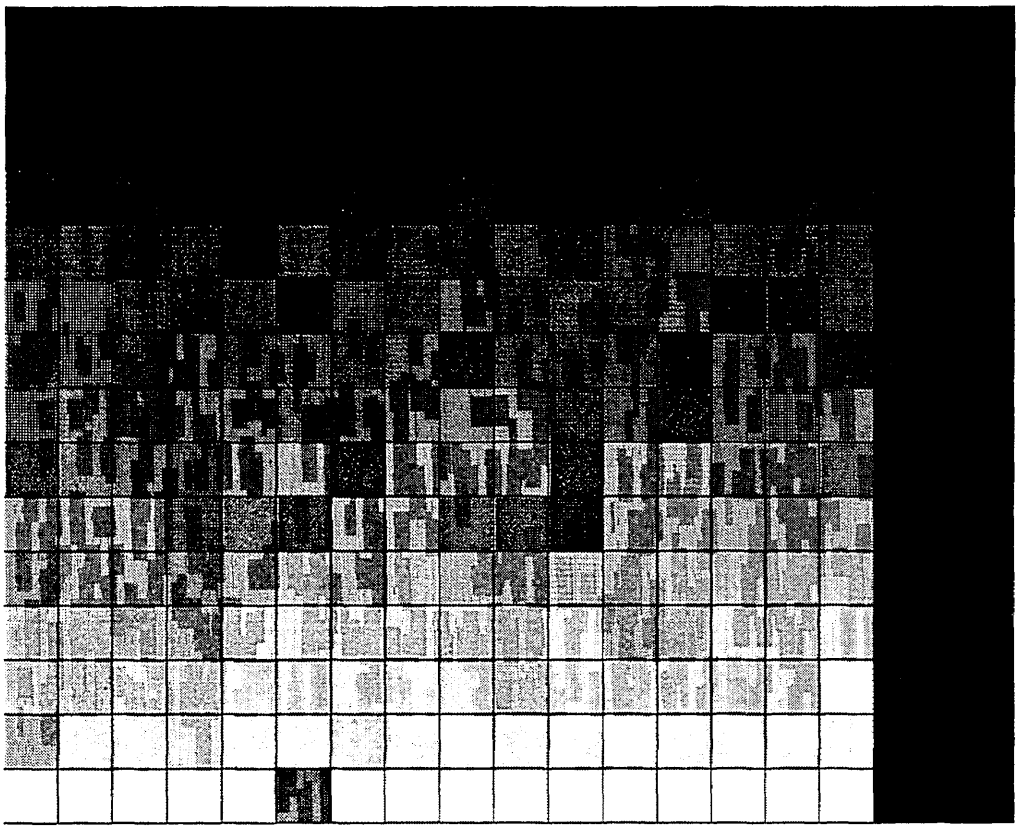


Fig. B49 — CAC color palette

*B10.4.3 Color Palette of a Palette Creation Thread*

After a color palette for a palette creation thread has been built, it can be viewed by selecting the *XXX_XX PALETTE* option from the *PLOT_PALETTE* selection (where XXX_XX is the palette thread). The Palette is displayed as described in Sec. B10.4.1.

*B10.4.4 Reviewing Downsampled Segments*

Downsampled segments that have been selected for generating a color palette are reviewed with the algebraic color palette before a new palette is created. For quality control purposes, the downsampled segments should be viewed with the new palette and verified for color accuracy *before* the new palette is inserted into the color palette database.

First, select the *DS SEGMENTS* option in the main menu bar. From the button cascade select either the *ALGEBRAIC, THREAD*, or *QUIT* option:

- If the palette has not yet been created, select the *ALGEBRAIC* option.
- If the palette has been created, select the *THREAD* option to review the segments with the new palette. If all segments are acceptable, the new palette can now be inserted into the color palette database.
- Select the "QUIT" button to exit CAC_QC.

The row and column numbers, latitude and longitude coordinates, and key name of a segment are displayed by moving the cursor to a particular segment and clicking button2. Clicking on another segment revises these values with information for that segment. A second click on the same segment removes this information from the screen.

# Appendix C

## CAC PROCESSING CHECK LIST

To ensure that the CPS is performing normally, there are several system checks that should be performed on a daily basis. These checks need only be performed when the CPS is actually processing data.

(1) Compare the DEFINE_xxx_LOGICALS.COM files stored in the MDFF_SYSTEM:[COMMON] directory to the Process_Chart programs currently executing in the MDFF_BATCH queue. *There should be a one-to-one correspondence.*

(2) Display the contents of the MDFF_BATCH queue and determine the following:

- Do all of the Process_Chart jobs that should be executing exist in the queue? If not, examine the end of the Process_Chart log file in the MDFF_SCRATCH:[000000] directory for that thread.
- The process named MDFF_CONTROL should also be executing (it maintains the CDROM reader database). If not, resubmit it to the queue:

  *submit/queue*=**MDFF_BATCH MDFFCOM:MDFF_CONTROL.**

(3) Execute *Pcode* to ensure that the correct processing thread is set. Use the *thread* command to set the correct processing thread. This is very important! ***Always*** check to make sure that current processing thread is correct. This is especially true when two threads of the same scale are being processed.

(4) Execute *List_Chart_Status* on each active thread. Ensure that the thread status is what is expected. For example, if a downsampling job has completed, then it should have progressed to the compression step.

(5) Review the PASS1 compressed data *as it completes*. Try not to get behind. This way, if there is a problem, it can be caught early and corrected.

(6) Execute *Sqcode* to display the contents of the compression queues. Ascertain that the generic queues are started. If not, execute the command:

*start/queue* **xxx_codebooks.**

This is an important check since the queue(s) may be halted, and without them, processing cannot proceed!

(7) All of the thread-specific command procedures and log files reside in that processing thread's MDFF_SCRATCH:[000000] directory. If there is a problem, look at the files in here first.

# Appendix D

## CAC PROCESSING SYSTEM STARTUP

This appendix describes the procedure used to start up the CPS:

Apply power to all of the hardware components and wait until the system console displays the ">>>" prompt.

The system console should be displaying the results of the power-on system checks. When everything appears acceptable, type *b* to start booting the system.

As the system boots, various messages will be displayed on the system console. These messages should be monitored to look for discrepant messages (i.e., messages that are not normally displayed during startup).

Once the system is booted:

- The logout statistics for the booting process are displayed on the operator console.
- Users can login to the system. The CPS *may be unavailable* for approximately 5 min after boot completion. This is due to the time required for the CPS to set up the CAC processing environment (i.e., thread-specific directories, logical names, queues, etc.). During this time, a process named SETUP_MDFF exists and will be displayed when the *show system* command is executed. This process will complete very shortly at which time the CPS is available and processing can resume.

During system startup, the MDFFEXE:STARTUP_MDFF.COM procedure is also submitted. This command procedure executes the above-mentioned MDFFEXE:SETUP_MDFF.COM procedure which sets up the CPS environment. The MDFFEXE:SETUP_MDFF.COM procedure creates the MDFF logical name table; mounts the various CPS disks; scans the MDFF_SYSTEM:[COMMON] directory for valid processing threads and configures them; and ensures that the necessary queues are available.

## Appendix E

## CAC PROCESSING SYSTEM (CPS) ACCEPTANCE TEST PROCEDURES

Each section in this document was designed to validate CPS capabilities. As such, this document was used in delivery of the CPS to NIMA. Since this information augments the *CAC Processing System Operators Manual,* there are references to sections in the manual for clarity and examples of use. All references to the *CAC Processing System Operators Manual* (this report) are denoted by the * symbol.

**I. CPS On-Line Help** (Test operation of the on-line CPS help library by accessing various topics)

A. Acronyms and Definitions
B. CAC Processing Overview (**Sec. 2.0*)
C. Source Code Documentation
D. Bitmap Definitions
E. Etc...
F. References: *MDFF HELP Library Version 2: On-Line Documentation for the Map Data Facility* and *MDFF HELP Library Design Document*

**II. CAC Processing System (CPS) Functions** (Test operation of each CPS phase)

A. INITIALIZATION *for a new CAC* (processing thread setup) (**Sec. 4.0*)
   1. Determine a scale of data to process (**Sec. 4.3*)
   2. Run Chart_Control (**Sec. 4.3*)
      Select the codebook queue to use (CODEBOOK1 or CODEBOOK2)
      Select NO for an update
      Enter the scale to process
      Enter a title for the new thread (a cover page will be printed)
      Verify that the thread is now in PASS1
   3. Use PLOT_ODI to determine ADRGs to include in new CAC
      (a) The generated file of ADRG selections is written to the current thread's MDFF_SCRATCH directory if a thread is set. Otherwise, the file is written to the current SYS$LOGIN directory. Note: DISPLAY_SEGMENTS_X only reads the ADRG selection file from the MDFF_SCRATCH directory.

B. INITIALIZATION *for a CAC update* (**Sec. 4.4*)
   1. Start a new thread as before, only select the update mode
   2. Use PLOT_ODI to determine ADRGs needed for the CAC update
   3. Print report from DTRLIB to get list of ADRG CDROMs in area of interest; ensure correctness
   4. Load the CAC to be updated on a CDROM reader

5. Run Chart_Control
   (a) Enter CDROM reader that the CAC is loaded on
6. Verify that the CAC is reloading:
   (a) $show system/subprocess
   (b) $directory CHART_ODI_DISK:[000000...]
   (c) Check the light on the CDROM reader
7. Continue as with a CAC build from scratch when the reload has completed

C. PASS1 (Downsampling and core segment compression) (**Sec. 5.0*)
1. Validate/Selection of ADRG CDROMs (**Sec. 5.2*)
   (a) Select an ADRG of the wrong scale (**Sec. 5.2.3*)
   (b) Select same ADRG CDROM twice
   (c) Verify correct selection with List_Chart_Status
2. ADRG data is copied to magnetic disk (**Sec. 5.2*)
3. Downsample ADRG into TS segments (**Sec. 5.3*)
   (a) Polar and nonpolar ADRGs
   (b) Ensure that the thread specific compression queue is started
4. Build compression procedures (**Sec. 5.4*)
5. Submit compression procedures
6. Monitor completion of compression procedures (**Sec. 3.5.4*)
   (a) List_Codebook_Status (**Sec. 3.4.7*)
   (b) Display_Segments_X (**Sec. 3.4.4*)
7. Review completed compression of an ADRG CDROM (CAC_QC) (**Sec. 5.6*)
8. Recover processing (**Sec. 5.7.3.1*)
   (a) Downsampling
   (b) Compression
   (c) Previous step(s)

D. PASS2 (Downsampling for edge segment filling) (**Sec. 6.0*)
1. Select CAC bounds (**Sec. 6.4*)
   (a) Verify CAC bounds with List_Odi_Bounds
2. Verify data was trimmed (Display_Segments_X) (**Sec. 6.6*)
3. Verify correct build of SEGMENTS file (Display_Segments_X)
4. Verify selection of ADRG CDROMs for PASS2 (**Sec. 6.7*)
5. Validate/Select PASS2 CDROMs (**Sec. 6.7*)
6. Verify downsampling of PASS2 CDROM(s)

E. PASS3 "A" (Compression of *filled edge* segments) (**Sec. 7.0*)
1. Build compression procedures (**Sec. 7.1–7.5*)
2. Submit compression procedures (**Sec. 7.1–7.5*)
3. Monitor completion of compression procedures
   (a) List_Chart_Status (**Sec. 3.4.6*)
   (b) List_Codebook_Status (**Sec. 3.4.7*)
   (c) Display_Segments_X (**Sec. 3.4.4*)
4. Review completed PASS3 "A" compression (CAC_QC)
5. Recover processing (**Sec. 7.9*)

F. PASS3 "B" (Compression of unfilled edge segments) (**Sec. 7.6*)
1. Review completed PASS3 "B" compression (CAC_QC)
2. Use CAC_QC to do a "Full Review." This verifies that there is no missing data

G. WRAPUP (**Sec. 8.0*)
1. Verify creation of ID directory files (Dump_CAC_Files and $dir)
2. Verify assigned CAC number and version (List_Chart_Status)
3. WRAPUP does extensive checking of the files in the CHART_ODI_DISK directories. A listing of the errors and CAC statistics is printed automatically. This must be checked before continuing

H. BUILD ISO IMAGE (**Sec. 9.0*)
1. Verify logging of CAC information in DTR
   (a) Run DTRLIB, use CAC_INFO option
2. Build the ISO image
3. Verify ISO image (CDDIR/FILE)
4. Copy ISO to 8-mm tape for mastering

## III. CAC Directories, Queues, and Command Procedures

A. Ensure the existence of the CPS directories (**Sec. 3.5*) and logical names (**Sec. 3.3*):
1. MDFFSRC: Source code
2. MDFFEXE: Executables and command files
3. Other MDFF_SYSTEM directories (BACKUP, COMMON)
4. MDFF_SCRATCH: Thread specific scratch area
5. CHART_SEGMENTS: Thread specific downsampled segments
6. CHART_ODI_DISK: Thread specific compressed segments
7. ADRGDIRxx: Scratch area for copied ADRG data

B. Ensure successful operation of the CPS queues (**Sec. 3.6*):
1. MDFF_BATCH (**Sec. 3.6.1*)
   (a) Process_Chart
   (b) Copy Procedures
   (c) Downsampling Procedures
2. Codebook Queues (Working queues for codebook compression) (**Sec. 3.6.2*)
   (a) CODEBOOK1
   (b) CODEBOOK2

## IV. Test CAC Utility Software

A. Processing thread utilities (**Sec. 3.1*):
1. Pcode: Returns the current processing thread name
2. List_Chart_Status: Lists the overall status of the current processing thread (**Sec. 3.4.6*)
3. List_Codebook_Status: Lists the status of current codebook compressions (**Sec. 3.4.7*)
4. Diskuse: Shows status of disk space on the disks assigned to processing
5. Sqcode: Shows the jobs running in the CODEBOOK queues
6. Sqmdff: Shows the jobs running in the MDFF_BATCH queue

B. CDROM reader utilities:
1. ADRGdirs: Directory listing of all devices assigned to store ADRG data
2. Reader_Space: Shows how much disk space is assigned to a given CDROM reader
3. Mount_CDROM: Mounts a CDROM on a specific CDROM reader
4. Dismount_CDROM: Dismounts a CDROM from a specific CDROM reader

C. Database and Quality Control Utilities:
1. DTRLIB: Main CAC/ADRG Database Control Program

2. Plot_ODI: Plots the locations of all ADRGs in the MDFF database, by series; CD coverages are color-coded to reflect whether they have been compressed into CAC (**Sec. 3.4.8*)
3. Display_Segments_X: Displays the segments in the current processing thread; each segment is color-coded to reflect its current status
4. CAC_QC: Quality control program for current processing thread (**Sec. 3.4.1*)

D. Miscellaneous Utilities:
1. CAC_Calc: Converts between (row, column), (latitude, longitude), and keyname for a given segment
2. Coors: Returns (latitude/longitude) of specified location from *MDFF_SYSTEM:[DATA]COORS.DAT*
3. Exam: Examines an ISO-8211 file, such as the Transmitter Header File on an ADRG CD
4. List_Odi_Bounds: Lists the geographic bounds (by zone) of the current thread
5. Dump_Cac_Files: Lists the contents of the non-image data files on a CAC or in the current thread (**Sec. 3.4.5*)
6. Cddir: Creates a directory list of an ODI image file; always use the /FILE qualifier (**Sec. 3.4.2*)

## V. Test Logging ADRG CDROMs into DTR Database

A. Ensure that all required database functions work correctly
B. Demonstrate DTRLIB

## VI. System Robustness/Error Handling

A. Select a "mismatched" ADRG CDROM for processing; ensure CPS handles this properly
B. Select a previously processed ADRG CDROM; test reprocessing (and omitting) it
C. Kill a DOWNSAMPLE job, then test *warm* and *cold* restart operations
D. Delete *all* compression jobs from a codebook queue, then ensure proper recovery
E. Regress processing from PASS3 to PASS1 to pick up another ADRG CDROM; test all stages
   (a) Note: Regression from WRAPUP is *not* supported
F. Kill processing during PASS3 (A and B) using the "kill" function of Chart_Control

## VII. CAC Color Palette Processing

A. Overview of CAC color palettes:
1. Color palette creation
2. Color palette database

B. Test all phases to create a new CAC color palette:
1. Initialize a palette processing thread with Palette_Control and demonstrate the similarities between palette and CAC processing
2. Select several segments from various ADRG CDROMs using DISPLAY_ADRG_X
3. Review downsampled segments using CAC_QC (algebraic) before continuing processing
4. Start PASS2 (MedianMedian) to create new palette
5. Review downsampled segments using CAC_QC (new palette) before continuing processing
6. Run WRAPUP to update the color palette database